

**TRƯỜNG ĐẠI HỌC BÁCH KHOA HÀ NỘI**

**ĐỒ ÁN TỐT NGHIỆP**  
**Sinh mô tả cho ảnh**

**TRỊNH THÀNH CÔNG**  
20165831@student.hust.edu.vn

**Ngành Công nghệ thông tin**

**Giảng viên hướng dẫn:** TS. Nguyễn Tuấn Dũng

\_\_\_\_\_  
Chữ ký của GVHD

**Bộ môn:** Khoa học máy tính

**Viện:** Công nghệ thông tin và truyền thông

**HÀ NỘI, 6/2020**

## PHIẾU GIAO NHIỆM VỤ ĐỒ ÁN TỐT NGHIỆP

### 1. Thông tin sinh viên

Họ và tên: **Trịnh Thành Công**

Email: trinhthanhcong98@gmail.com

Điện thoại liên lạc: **0967452598**

Hệ đào tạo: **Đại học chính quy**

Lớp: **CN-CNTT-TT.01-K61**

Đồ án tốt nghiệp được thực hiện tại: Bộ môn Khoa học máy tính, Viện Công nghệ thông tin và truyền thông, Đại học Bách Khoa Hà Nội.

Thời gian làm ĐATN: Từ ngày 16/03/2020 đến 26/06/2020

### 2. Mục đích của đồ án tốt nghiệp

Nghiên cứu và áp dụng mô hình mạng nơ-ron tích chập và mạng nơ-ron hồi quy trong bài toán sinh mô tả cho ảnh.

### 3. Các nhiệm vụ cụ thể của ĐATN

- 1) Giới thiệu bài toán sinh mô tả cho ảnh
- 2) Nghiên cứu cơ sở lí thuyết mạng nơ-ron tích chập và mạng nơ-ron hồi quy
- 3) Xây dựng mô hình giải quyết bài toán sinh mô tả cho ảnh
- 4) Thử nghiệm và đánh giá
- 5) Kết luận và hướng phát triển

### 4. Lời cam đoan của sinh viên

Tôi - Trịnh Thành Công – cam kết ĐATN là công trình nghiên cứu của bản thân tôi dưới sự hướng dẫn của TS. Nguyễn Tuấn Dũng.

Các kết quả nêu trong ĐATN là trung thực, không phải là sao chép toàn văn của bất kỳ công trình khác.

### 5. Xác nhận của giáo viên hướng dẫn về mức độ hoàn thành của ĐATN và cho phép bảo vệ

.....  
.....

Hà Nội, ngày      tháng 06 năm 2020

Giáo viên hướng dẫn

## LỜI CẢM ƠN

Em xin gửi cảm ơn chân thành nhất tới TS. Nguyễn Tuấn Dũng, bộ môn Khoa học máy tính, Viện công nghệ thông tin – truyền thông, trường Đại học Bách Khoa Hà Nội đã gợi ý đề tài, tận tình hướng dẫn giúp em hoàn thành tốt đồ án này. Phong cách làm việc, giảng dạy, truyền đạt kiến thức của Thầy đã tạo động lực, cảm hứng cho em và rất nhiều các bạn sinh viên khác.

Em xin cảm ơn tất cả các thầy cô giảng viên của trường Đại học Bách Khoa Hà Nội đặc biệt là thầy cô thuộc Viện công nghệ thông tin – truyền thông đã trang bị những kiến thức, kỹ năng vô cùng bổ ích và quý giá trong suốt các năm học vừa qua. Những kiến thức, kỹ năng đó sẽ là hành trang vững chắc cho em bước vào tương lai.

Cuối cùng, con xin cảm ơn bố mẹ, các anh chị em đã luôn quan tâm, động viên, chăm sóc, tạo điều kiện cho con trong quá trình học tập tại trường.

Mặc dù đã cố gắng hết sức để hoàn thành đồ án tốt nghiệp, nhưng chắc chắn sẽ không tránh khỏi những sai sót. Mong nhận được sự góp ý chân thành từ các quý thầy cô và các bạn.

## TÓM TẮT NỘI DUNG ĐỒ ÁN TỐT NGHIỆP

Đồ án được tổ chức thành các phần với các nội dung cụ thể như sau:

**Mở đầu:** Trình bày về bài toán sinh mô tả cho ảnh, ứng dụng của bài toán trong thực tiễn.

**Chương 1: Cơ sở lý thuyết:** Chương này sẽ đi sâu vào tìm hiểu lý thuyết mạng nơ-ron nhân tạo, mạng nơ-ron tích chập, mạng nơ-ron hồi quy, là chìa khóa giải quyết bài toán sinh mô tả ảnh.

**Chương 2: Mô hình sinh mô tả ảnh với cơ chế Chú ý:** Mô tả tổng quan hệ thống, chi tiết mô hình với cơ chế Chú ý.

**Chương 3: Cài đặt chi tiết và đánh giá:** Môi trường cài đặt, các bước tiền xử lý dữ liệu, triển khai mô hình, quá trình huấn luyện, kết quả thực nghiệm. Từ đó, ta sẽ đưa ra một số phân tích, đánh giá, so sánh và kết luận về điểm mạnh và điểm hạn chế của phương pháp.

**Kết luận và hướng phát triển:** Cuối cùng là sẽ tổng kết các nội dung đã trình bày trong đồ án, từ đó đề xuất các phương hướng nghiên cứu tiếp theo để tiếp tục cải thiện chất lượng của hệ thống.

*Hà Nội, ngày 26 tháng 06 năm 2020*

Tác giả ĐATN

## MỤC LỤC

MỤC LỤC .....	4
MỞ ĐẦU .....	10
CHƯƠNG 1: CƠ SỞ LÝ THUYẾT .....	11
1.1 Học sâu.....	11
1.1.1 Giới thiệu về học sâu .....	11
1.1.2 Mạng nơ-ron.....	11
1.2 Mạng nơ-ron tích chập.....	14
1.2.1 Lớp tích chập.....	15
1.2.2 Lớp gộp .....	16
1.2.3 Tầng kết nối đầy đủ .....	17
1.2.4 Một số mạng nơ-ron tích chập hiện đại .....	17
1.3 Mạng Hồi Quy.....	21
1.3.1 Tổng quan.....	21
1.3.2 Mạng LSTM .....	23
1.3.3 Mạng GRU .....	26
1.4 Xử lý ngôn ngữ tự nhiên.....	27
1.4.1 Tổng quan về xử lý ngôn ngữ tự nhiên.....	27
1.4.2 Biểu diễn từ và văn bản.....	28
CHƯƠNG 2: MÔ HÌNH SINH MÔ TẢ ẢNH.....	30
2.1 Tổng quan hệ thống .....	30
2.1.1 Mô hình sinh mô tả ảnh .....	30
2.1.2 Kiến trúc Encoder-Decoder cho bài toán sinh mô tả ảnh .....	31
2.2 Mô hình sinh mô tả ảnh đề xuất .....	32
2.2.1 Đề xuất cơ chế Chú ý trong bài toán sinh mô tả ảnh.....	32
2.2.2 Đề xuất tìm kiếm chùm.....	34
2.2.3 Chi tiết mô hình .....	34
CHƯƠNG 3: CÀI ĐẶT CHI TIẾT VÀ ĐÁNH GIÁ .....	38
3.1 Môi trường cài đặt mô hình và công cụ sử dụng .....	38
3.1.1 Môi trường cài đặt mô hình.....	38
3.1.2 Công cụ sử dụng .....	38

3.2 Thu thập dữ liệu .....	39
3.3 Tiền xử lí dữ liệu ảnh.....	39
3.4 Tiền xử lí mô tả .....	41
3.5 Xây dựng mô hình .....	42
3.6 Huấn luyện mô hình .....	42
3.7 Kết quả thực nghiệm.....	44
3.7.1 Độ đo.....	45
3.7.2 Kết quả đánh giá trên bộ dữ liệu Flick8k.....	46
3.7.3 Một vài kết quả từ bộ dữ liệu .....	47
3.8 Đánh giá mô hình .....	50
CHƯƠNG 4: KẾT LUẬN VÀ ĐỊNH HƯỚNG PHÁT TRIỂN .....	51
4.1 Kết luận.....	51
4.2 Định hướng phát triển.....	51
DANH MỤC TÀI LIỆU THAM KHẢO .....	52

## DANH MỤC TỪ VIẾT TẮT VÀ THUẬT NGỮ

STT	Tên viết tắt	Tên đầy đủ	Ý nghĩa
1	ANN	Artificial Neural Network	Mạng nơ-ron nhân tạo
2	CNN	Convolution Neural Network	Mạng nơ-ron tích chập
3	RNN	Recurrent Neural Network	Mạng nơ-ron hồi quy
4	LSTM	Long Short Term Memory	Mạng nơ-ron bộ nhớ ngắn hạn
5	GRU	Gated Recurrent Units	Mạng nơ-ron các đơn vị cổng hồi quy
6	Encoder-Decoder	Encoder-Decoder	Mã hóa – Giải mã
7	Computer Vision	Computer Vision	Thị giác máy tính
8	Attention	Attention	Cơ chế chú ý

## DANH MỤC HÌNH ẢNH

Hình 1.1- Mối quan hệ giữa trí tuệ nhân tạo, học máy và học sâu.....	11
Hình 1.2- Cấu trúc của một mạng nơ-ron thông thường.....	12
Hình 1.3- So sánh giữa hàm kích hoạt tuyến tính và hàm kích hoạt phi tuyến.....	12
Hình 1.4- Các hàm kích hoạt phi tuyến thường dùng trong học sâu .....	13
Hình 1.5- Minh họa về tốc độ học .....	13
Hình 1.6- Giải thuật lan truyền ngược trong mạng nơ-ron đa tầng .....	14
Hình 1.7- Minh họa về các filter trong mạng nơ-ron tích chập.....	15
Hình 1.8- Ví dụ minh họa về padding.....	16
Hình 1.9- Ví dụ minh họa về toán tử pooling.....	16
Hình 1.10- Max pooling và Average pooling.....	17
Hình 1.11- Tầng kết nối đầy đủ .....	17
Hình 1.12- Kiến trúc mạng VGG.....	18
Hình 1.13- Tham số trong mạng VGG.....	19
Hình 1.14- Khối Inception thông thường .....	19
Hình 1.15- Khối Inception cải tiến .....	20
Hình 1.16- Minh họa ý tưởng của mạng ResNet.....	20
Hình 1.17- Kiến trúc mạng ResNet.....	21
Hình 1.18- Cấu trúc của một mạng hồi quy thông thường .....	22
Hình 1.19- Bảng chuyển cell state trong LSTM.....	23
Hình 1.20- Cấu trúc của mạng LSTM.....	24
Hình 1.21- Cổng quên .....	25
Hình 1.22- Cổng vào .....	25
Hình 1.23- Cập nhật trạng thái cell state.....	26
Hình 1.24- Cổng ra.....	26
Hình 1.25- LSTM giải quyết vấn đề vanishing gradient.....	26
Hình 1.26- Cấu trúc của mạng GRU .....	27
Hình 1.27- Các lĩnh vực trong xử lý ngôn ngữ tự nhiên.....	27
Hình 1.28- Ví dụ minh họa về Word2Vec .....	28
Hình 2.1- Trích xuất đặc trưng .....	30
Hình 2.2- Mô hình ngôn ngữ .....	31
Hình 2.3- Mô hình Encoder-Decoder cho bài toán sinh mô tả ảnh.....	32
Hình 2.4- Mô hình Attention.....	33
Hình 2.5 – Giải thuật tìm kiếm chùm trong dịch máy .....	34
Hình 2.6- Mô hình sinh mô tả ảnh với cơ chế Attention .....	35
Hình 2.7- Minh họa Encoder CNN.....	36
Hình 2.8- Mạng LSTM với cơ chế chú ý.....	36
Hình 3.1- Điểm tăng trưởng của các thư viện deep learning trong năm 2019.....	38

Hình 3.2- Dữ liệu dùng để huấn luyện mô hình .....	39
Hình 3.3- Ảnh sau khi đã resize về kích thước (299, 299,3).....	40
Hình 3.4- Mô hình trích xuất đặc trưng (Inception v3) .....	40
Hình 3.5- Dữ liệu trước khi được làm sạch.....	41
Hình 3.6- Dữ liệu sau khi được làm sạch.....	41
Hình 3.7- Top 50 từ có tần suất xuất hiện nhiều nhất trong dataset .....	42
Hình 3.8- Thông số của encoder.....	43
Hình 3.9- Thông số mô hình Attention .....	43
Hình 3.10- Thông số mô hình Decoder (RNN) với cơ chế Attention .....	44
Hình 3.11- Thông số mô hình.....	44
Hình 3.12- Giá trị của hàm loss function qua thời gian trên tập train và validate .....	45
Hình 3.13- Một vài kết quả từ bộ dữ liệu (1) .....	48
Hình 3.14- Một vài kết quả thu được từ bộ dữ liệu (2).....	48
Hình 3.15- Một vài kết quả thu được từ bộ dữ liệu (3).....	49
Hình 3.16- Một vài kết quả thu được từ bộ dữ liệu (4).....	49
Hình 3.17- Một vài kết quả thu được từ bộ dữ liệu (5).....	50



## DANH MỤC BẢNG

Bảng 3.1- Trọng số của n-grams ứng với các loại điểm BLEU khác nhau .....	47
Bảng 3.2- Số điểm BLEU trên tập dữ liệu Flickr8k test .....	47
Bảng 3.3- Điểm BLEU trên tập dữ liệu Flickr8k test do đại học stanford thực hiện .....	47

# MỞ ĐẦU

## 1. Giới thiệu đề tài

Sinh mô tả cho ảnh là một nhiệm vụ rất gần với khả năng hiểu bức ảnh – một trong những nhiệm vụ chính của Thị giác máy tính. Mô hình không những phải xác định có những đối tượng nào trong một bức ảnh mà còn phải có khả năng biểu diễn mối quan hệ giữa các đối tượng qua một ngôn ngữ nào đó. Do đó, nhiệm vụ này từ lâu đã được xem là một vấn đề khó khăn đối với các thuật toán học máy, bởi vì nó đòi hỏi mô hình phải bắt trước được khả năng vượt trội của con người đó là trích xuất những thông tin nổi bật từ bức ảnh, diễn đạt ra một ngôn ngữ nào đó.

Tuy nhiên trong thời gian gần đây, với khả năng tính toán vượt trội của máy tính và lượng dữ liệu khổng lồ của con người tạo ra, chúng ta đang có những đột phá thực sự trong lĩnh vực học máy, và đặc biệt là học sâu. Chính vì những lí do đó, nên em đã chọn đề tài “Sinh mô tả cho ảnh” trong đồ án tốt nghiệp của mình. Đây cũng là cơ hội quý giá để em trang bị thêm kiến thức về học sâu, cùng với việc tiếp tục tìm tòi học hỏi để tạo ra những ứng dụng có ích cho xã hội.

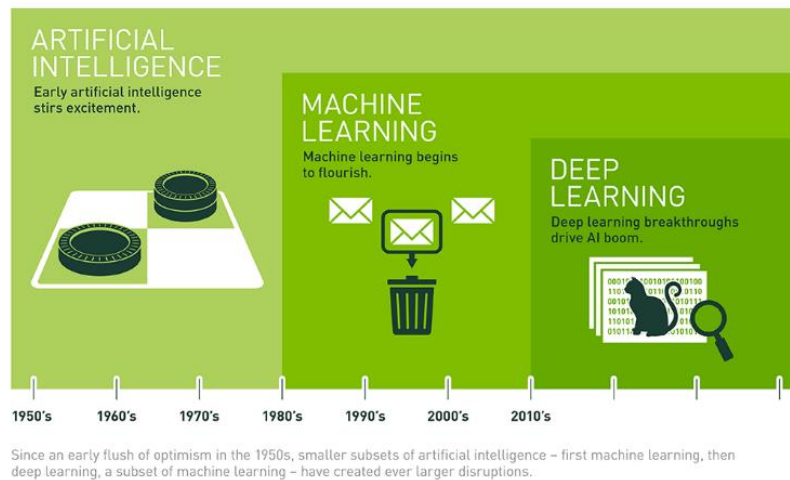
## 2. Ứng dụng của sinh mô tả cho ảnh trong thực tế

- Lái xe tự hành: Đây là một trong những thách thức lớn nhất trong ngành Trí tuệ nhân tạo. Nếu chúng ta có những mô tả phù hợp cho những cảnh vật xung quanh chiếc xe, nó có thể mang lại một bước tiến cho hệ thống lái xe tự hành.
- Trợ giúp người mù: Chúng ta có thể tạo ra một sản phẩm cho người mù, giúp họ đi trên đường mà không cần sự giúp đỡ của bất cứ ai khác. Chúng ta có thể làm điều đó bằng cách chuyển khung cảnh thành văn bản rồi từ văn bản chuyển thành giọng nói.
- Camera quan sát: Camera quan sát ở khắp mọi nơi, nhưng cùng với việc quan sát, liệu rằng chúng ta có thể tạo ra những mô tả tương ứng để cảnh báo một vụ trộm hoặc một hoạt động nguy hiểm nào đó.
- Truy vấn cơ sở dữ liệu hình ảnh: chúng ta có thể tìm những hình ảnh liên quan đến nội dung nào đó trong một tập cơ sở dữ liệu hình ảnh lớn.

# CHƯƠNG 1: CƠ SỞ LÝ THUYẾT

## 1.1 Học sâu

### 1.1.1 Giới thiệu về học sâu



Hình 1.1- Mối quan hệ giữa trí tuệ nhân tạo, học máy và học sâu

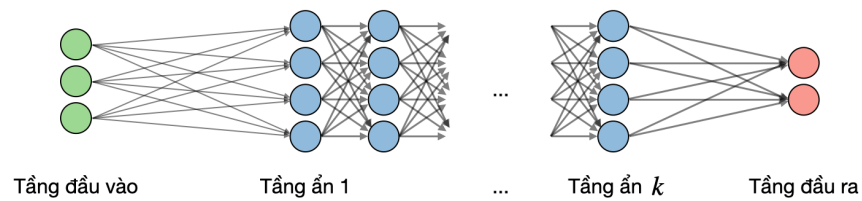
Học sâu là một ngành nhỏ của học máy, được bắt nguồn từ thuật toán trong mạng nơ-ron đa lớp. Phương pháp học máy truyền thống đòi hỏi trích xuất đặc trưng một cách thủ công, đòi hỏi kinh nghiệm và phụ thuộc vào từng bài toán cụ thể. Khắc phục được nhược điểm này, học sâu cho phép trích chọn đặc trưng một cách tự động từ dữ liệu. Học sâu có thể xử lý dữ liệu theo cách tương tự như bộ não con người có thể thực hiện. Điểm khác biệt ở đây là chúng ta sẽ không phải dạy một mô hình học sâu biết một con mèo trông như thế nào mà chỉ cần cung cấp cho mô hình đủ hình ảnh cần thiết về loài mèo, và mô hình sẽ tự mình hình dung ra được.

### 1.1.2 Mạng nơ-ron

Mạng nơ-ron là các mô hình mô phỏng lại mạng nơ-ron sinh học. Nó là một cấu trúc khối gồm các đơn vị tính toán đơn giản được liên kết chặt chẽ với nhau, trong đó các liên kết giữa các nơ-ron quyết định chức năng của mạng. Các loại mạng nơ-ron thường được sử dụng bao gồm: Mạng nơ-ron tích chập (CNN) và mạng nơ-ron hồi quy (RNN)

#### a. Kiến trúc :

Các thuật ngữ xoay quanh kiến trúc của mạng nơ-ron được mô tả bằng hình phía dưới:



Hình 1.2- Cấu trúc của một mạng nơ-ron thông thường

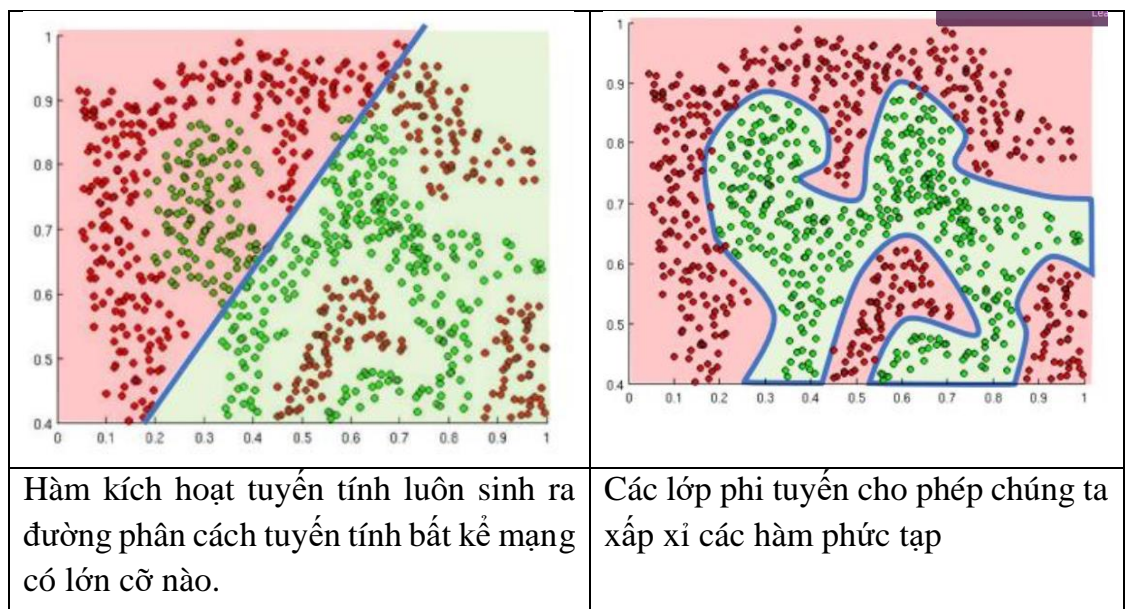
Kí hiệu  $i$  là tầng thứ  $i$  trong mạng,  $j$  là đơn vị ẩn (hidden unit) thứ  $j$  của tầng, ta có:

$$z_j^{(i)} = w_j^{(i)} x + b_j^{(i)}$$

Trong đó  $w$ ,  $b$ ,  $z$  tương ứng với trọng số (weight), bias và đầu vào.

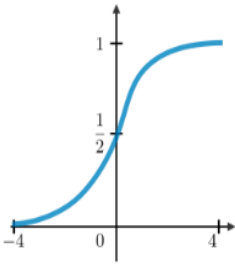
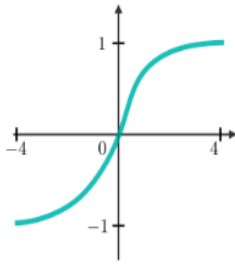
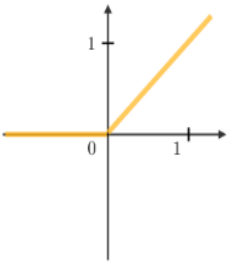
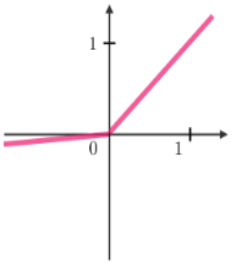
### b. Hàm kích hoạt

Hàm kích hoạt thường được sử dụng ở phần cuối của các đơn vị ẩn (hidden units) nhằm mục đích đưa các lớp phi tuyến vào mạng nơ-ron.



Hình 1.3- So sánh giữa hàm kích hoạt tuyến tính và hàm kích hoạt phi tuyến

Dưới đây là những hàm kích hoạt phổ biến nhất:

Sigmoid	Tanh	ReLU	Leaky ReLU
$g(z) = \frac{1}{1 + e^{-z}}$	$g(z) = \frac{e^z - e^{-z}}{e^z + e^{-z}}$	$g(z) = \max(0, z)$	$g(z) = \max(\epsilon z, z)$ với $\epsilon \ll 1$
			

Hình 1.4- Các hàm kích hoạt phi tuyến thường dùng trong học sâu

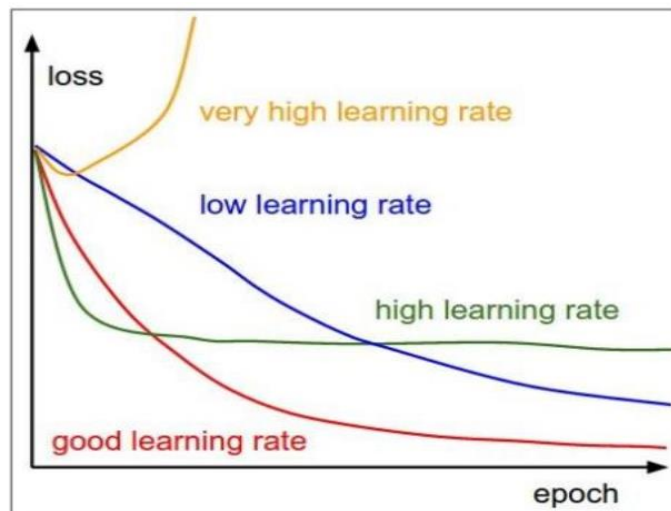
### c. Hàm lỗi

Trong mạng nơ-ron, hàm lỗi cross-entropy  $L(z, y)$  giữa 2 phân phối  $y$  và  $z$  thường được sử dụng và định nghĩa như sau:

$$L(z, y) = -[y \log(z) + (1-y) \log(1-z)]$$

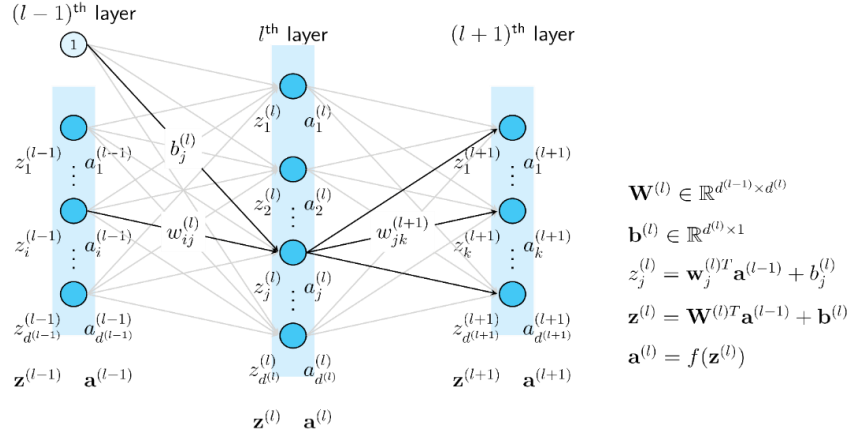
### d. Tốc độ học

Tốc độ học thường được kí hiệu là  $\alpha$  hoặc  $\eta$ , chỉ ra tốc độ mà trọng số được cập nhật. Thông số này có thể là cố định hoặc được thay đổi tùy biến. Thường bắt đầu với giá trị lớn và giảm dần theo thời gian.



Hình 1.5- Minh họa về tốc độ học

### e. Giải thuật lan truyền ngược



Hình 1.6- Giải thuật lan truyền ngược trong mạng nơ-ron đa tầng

Là giải thuật để cập nhật trọng số trong mạng nơ-ron bằng cách tập trung vào sự khác nhau giữa đầu ra thực tế và đầu ra mong muốn. Đạo hàm của hàm mất mát với trọng số  $\mathbf{W}$  được tính bằng quy tắc chain rule theo như cách dưới đây:

$$\frac{\partial L(z, y)}{\partial \mathbf{W}} = \frac{\partial L(z, y)}{\partial \mathbf{a}} \times \frac{\partial \mathbf{a}}{\partial \mathbf{z}} \times \frac{\partial \mathbf{z}}{\partial \mathbf{w}}$$

Trọng số được cập nhật:

$$\mathbf{W} \leftarrow \mathbf{W} - \alpha \frac{\partial L(z, y)}{\partial \mathbf{W}}$$

## 1.2 Mạng nơ-ron tích chập

Các kiến trúc dựa trên mạng nơ-ron tích chập hiện nay xuất hiện trong mọi ngóc ngách của lĩnh vực thị giác máy tính, và đã trở thành kiến trúc chủ đạo cho các ứng dụng thương mại hay một mô hình dùng để tham gia một cuộc thi nào đó liên quan tới nhận dạng ảnh, phát hiện đối tượng, hay phân vùng theo ngữ cảnh,...

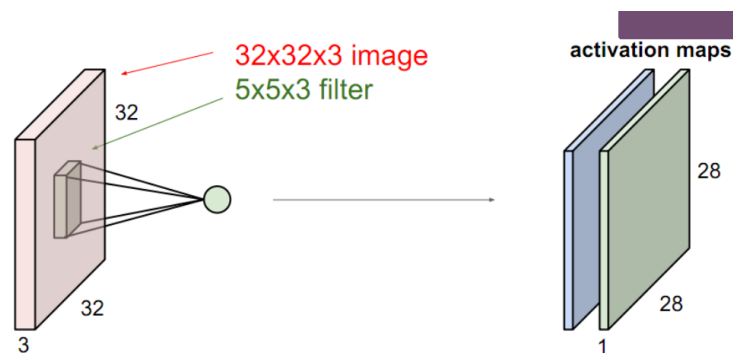
Các mạng nơ-ron tích chập thường có hiệu quả tính toán cao hơn các kiến trúc mạng dày đặc bởi vì đòi hỏi ít tham số hơn và dễ thực thi song song trên nhiều GPU. Do đó, các mạng nơ-ron tích chập đã nhanh chóng trở thành một công cụ quan trọng và tin cậy thậm chí cả với những tác vụ liên quan đến cấu trúc tuần tự

một chiều như là xử lý âm thanh, văn bản và phân tích chuỗi thời gian mà ở đó các mạng nơ-ron hồi quy thường được sử dụng.

### 1.2.1 Lớp tích chập

Nếu sử dụng mạng nơ-ron kết nối đầy đủ, khi kích thước ảnh tăng lên thì số lượng tham số của mô hình cũng tăng lên rất lớn. Do đó, khó có thể sử dụng mạng nơ-ron kết nối đầy đủ trong thực tế. Hơn nữa, trong ảnh các pixel ở cạnh nhau thường có liên kết, chia sẻ thông tin với nhau hơn là những pixel ở xa nên người ta đã đề xuất ra mạng nơ-ron tích chập.

Khác với nơ-ron kết nối đầy đủ, mỗi nơ-ron tích chập (filter) chỉ kết nối cục bộ với dữ liệu đầu vào. Nơ-ron tích chập trượt từ trái sang phải và từ trên xuống dưới khối dữ liệu đầu vào và tính toán bằng phép nhân (element-wise) để sinh ra một bản đồ kích hoạt (activation map). Chiều sâu của nơ-ron tích chập bằng chiều sâu của khối dữ liệu đầu vào.



Hình 1.7- Minh họa về các filter trong mạng nơ-ron tích chập

Dữ liệu đi qua tầng tích chập thì kích thước sẽ bị giảm đi theo công thức:

Output size:

$$(N - F) / \text{stride} + 1$$

Trong đó  $N*N$  là kích thước đầu vào,  $F*F$  là kích thước của nơ-ron tích chập và stride là bước nhảy.

Trong một số trường hợp, ta muốn bảo toàn kích thước đầu ra thì sẽ thêm viền bởi các số 0 (zero padding).

0	0	0	0	0	0			
0								
0								
0								
0								

Hình 1.8- Ví dụ minh họa về padding

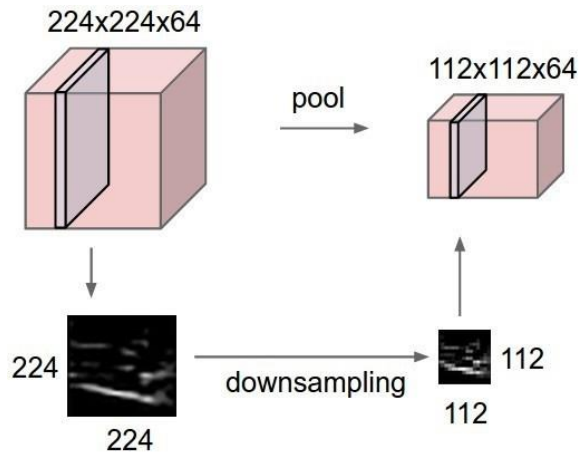
Ví dụ hình trên, với đầu vào kích thước  $7 \times 7$ , nơ-ron kích thước  $3 \times 3$ , bước nhảy stride 1, padding viền độ rộng 1 thì khi đó kích thước đầu ra sẽ là  $7 \times 7$ .

### 1.2.2 Lớp gộp

Khi xử lý ảnh, ta thường muốn giảm dần độ phân giải của khối dữ liệu để giảm bộ nhớ và khối lượng tính toán. Trong mạng nơ-ron tích chập, lớp gộp sẽ thực hiện nhiệm vụ này. Ngoài ra, lớp gộp giúp mạng biểu diễn bất biến đối với các thay đổi tịnh tiến (translation invariance) hoặc biến dạng (deformation invariance) của dữ liệu đầu vào.

Gọi pooling size kích thước  $K \times K$ . Đầu vào của lớp gộp có kích thước  $H \times W \times D$ , ta tách ra làm  $D$  ma trận kích thước  $H \times W$ . Với mỗi ma trận, trên vùng kích thước  $K \times K$  trên ma trận ta tìm maximum hoặc average của dữ liệu rồi viết vào ma trận kết quả. Quy tắc về stride và padding áp dụng như phép tích chập trên ảnh.

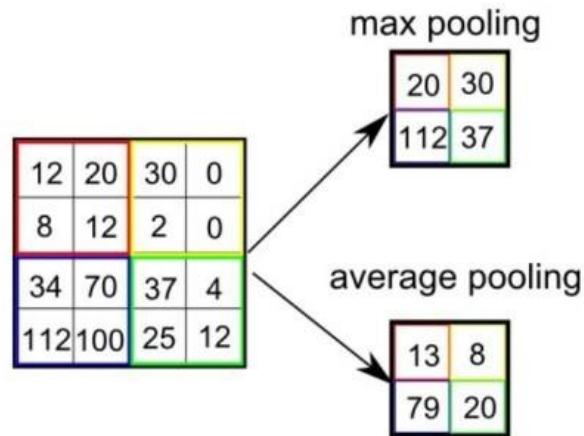
Trong thực tế, khi dùng tầng gộp thì người ta sẽ dùng size = (2, 2), stride = 2 và padding = 0. Khi đó chiều dài và rộng của dữ liệu giảm đi một nửa, còn chiều sâu thì được giữ nguyên.



Hình 1.9- Ví dụ minh họa về toán tử pooling



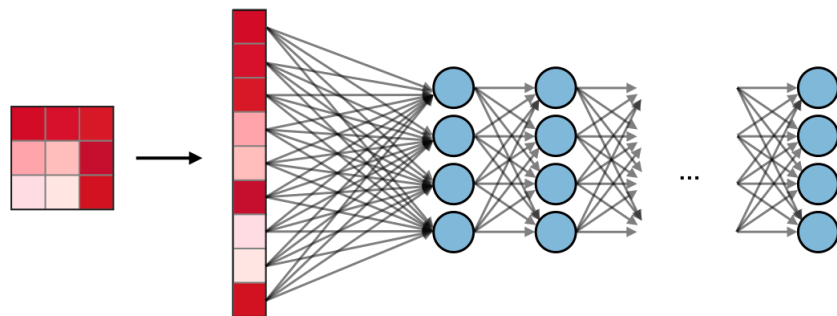
Có nhiều toán tử pooling, nhưng max pooling và average pooling là những dạng pooling hay được sử dụng, tương ứng với nó là giá trị lớn nhất và giá trị trung bình được lấy ra.



Hình 1.10- Max pooling và Average pooling

### 1.2.3 Tầng kết nối đầy đủ

Tầng kết nối đầy đủ nhận đầu vào là các dữ liệu đã được làm phẳng, mà mỗi đầu vào đó được kết nối đến tất cả nơ-ron. Trong mô hình mạng CNNs, các tầng kết nối đầu đủ thường ở cuối mạng và được dùng để tối ưu hóa mục tiêu của mạng ví dụ như độ chính xác của lớp.



Hình 1.11- Tầng kết nối đầy đủ

### 1.2.4 Một số mạng nơ-ron tích chập hiện đại

#### a. Mạng VGGnet

Ra đời vào năm 2014, mạng VGG là biến thể của các mạng tích chập truyền thống nhưng mạng sâu hơn với 16 -19 tầng, cùng với số lượng tham số khổng lồ là 140 triệu tham số.



Hình 1.12- Kiến trúc mạng VGG

Mạng VGG sử dụng nơ-ron có kích thước nhỏ, chỉ dùng conv 3\*3, bước nhảy 1, padding 1 và tầng gộp 2\*2 max pool, bước nhảy 2. Ở đây VGG chèn 3 lớp tích chập 3\*3 conv (stride 1) có hiệu quả tương đương với một lớp 7\*7 conv, nhưng mạng sâu hơn và nhiều lớp phi tuyến hơn, cùng với đó là chi phí tính toán thấp do ít tham số hơn.

Sau các tầng tích chập và tầng gộp là đến 3 tầng kết nối đầy đủ: 2 tầng đầu có kích thước 4096, còn tầng cuối có kích thước là 1000 tương ứng với nhiệm vụ phân loại 1000 nhãn khác nhau trong cuộc thi ImageNet.

Input	memory: 224*224*3=150K	params: 0
3x3 conv, 64	memory: 224*224*64=3.2M	params: (3*3*3)*64 = 1,728
3x3 conv, 64	memory: 224*224*64=3.2M	params: (3*3*64)*64 = 36,864
Pool	memory: 112*112*64=800K	params: 0
3x3 conv, 128	memory: 112*112*128=1.6M	params: (3*3*64)*128 = 73,728
3x3 conv, 128	memory: 112*112*128=1.6M	params: (3*3*128)*128 = 147,456
Pool	memory: 56*56*128=400K	params: 0
3x3 conv, 256	memory: 56*56*256=800K	params: (3*3*128)*256 = 294,912
3x3 conv, 256	memory: 56*56*256=800K	params: (3*3*256)*256 = 589,824
3x3 conv, 256	memory: 56*56*256=800K	params: (3*3*256)*256 = 589,824
Pool	memory: 28*28*256=200K	params: 0
3x3 conv, 512	memory: 28*28*512=400K	params: (3*3*256)*512 = 1,179,648
3x3 conv, 512	memory: 28*28*512=400K	params: (3*3*512)*512 = 2,359,296
3x3 conv, 512	memory: 28*28*512=400K	params: (3*3*512)*512 = 2,359,296
Pool	memory: 14*14*512=100K	params: 0
3x3 conv, 512	memory: 14*14*512=100K	params: (3*3*512)*512 = 2,359,296
3x3 conv, 512	memory: 14*14*512=100K	params: (3*3*512)*512 = 2,359,296
3x3 conv, 512	memory: 14*14*512=100K	params: (3*3*512)*512 = 2,359,296
Pool	memory: 7*7*512=25K	params: 0
FC 4096	memory: 4096	params: 7*7*512*4096 = 102,760,448
FC 4096	memory: 4096	params: 4096*4096 = 16,777,216
FC 1000	memory: 1000	params: 4096*1000 = 4,096,000

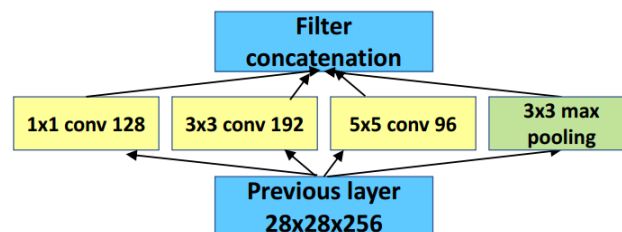
Hình 1.13- Tham số trong mạng VGG

## b. Mạng GoogleNet

Năm 2014, các nhà nghiên cứu của google đã đưa mạng GoogleNet tham dự cuộc thi ImageNet và kết quả mạng GoogleNet đã trở thành nhà vô địch trong tác vụ phân loại ảnh.

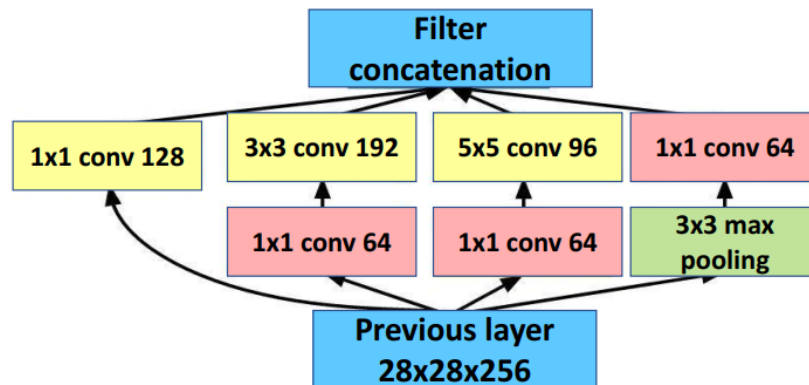
Mô hình này rất đặc biệt, mô hình có 22 lớp bao gồm các khối Inception và không có lớp kết nối đầy đủ. Google Net chỉ có 5 triệu tham số, vì mục tiêu của nó là tập trung vào giảm độ phức tạp tính toán.

Với mạng tích chập thông thường, khi thiết kế phải bắt buộc xác định trước các tham số của một lớp tích chập như là: kernel\_size, padding, strides,... Thường rất khó để xác định tham số nào là phù hợp. Khối Inception trong mạng GoogleNet được tạo ra để giải quyết vấn đề này. Thay vì chỉ sử dụng một tầng tích chập với các tham số cố định, ta hoàn toàn có thể sử dụng cùng một lúc nhiều lớp tích chập khác nhau, sau đó được tổng hợp lại thành đầu ra cho khối Inception.



Hình 1.14- Khối Inception thông thường

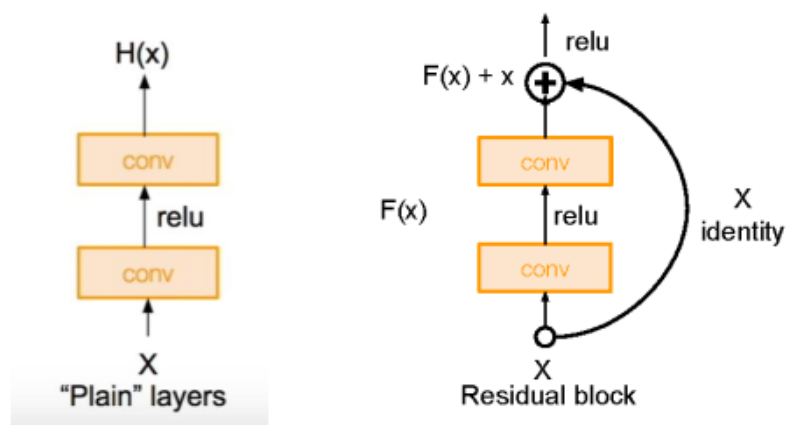
Để giảm chi phí tính toán, Khối Inception đã sử dụng các tầng tích chập kích thước  $1 \times 1$  để giảm chiều sâu của đầu vào. Ngoài ra, còn có ưu điểm là tăng các lớp phi tuyến cho mạng.



Hình 1.15- Khối Inception cải tiến

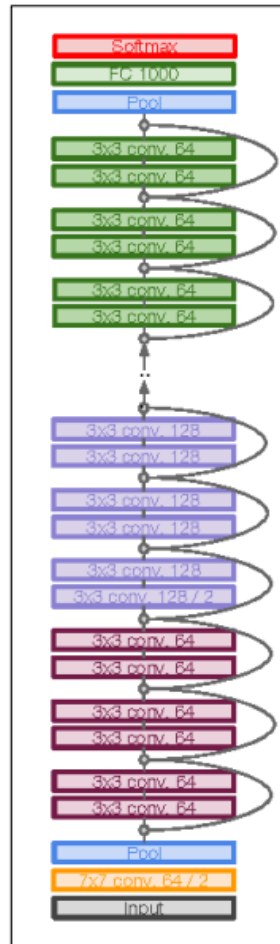
### c. Mạng ResNet

ResNet được phát triển bởi microsoft năm 2015 với bài báo “ Deep residual learning for image recognition”. Vấn đề khi ta huấn luyện các mô hình tích chập ta thường gặp vấn đề triệt tiêu hoặc bùng nổ đạo hàm. Thực tế cho thấy khi số lượng tầng trong mạng tích chập tăng thì độ chính xác của mô hình tăng theo, tuy nhiên khi tăng số tầng quá lớn thì độ chính xác lại bị giảm đi. Lí do ở đây không phải là mô hình không đủ tốt, mà là vấn đề ở bài toán tối ưu. Do đó mạng ResNet ra đời với ý tưởng học biểu diễn phần dư (sự sai khác giữa đầu ra và đầu vào) thay vì học trực tiếp đầu ra như trước.



Hình 1.16- Minh họa ý tưởng của mạng ResNet

Mạng ResNet gồm các khối phần dư (residual blocks) xếp chồng lên nhau, mỗi khối có hai lớp tích chập  $3 \times 3$ . Theo định kì, số lượng bộ lọc (filter) được tăng gấp đôi và giảm độ phân giải bằng lớp tích chập với bước nhảy bằng 2.



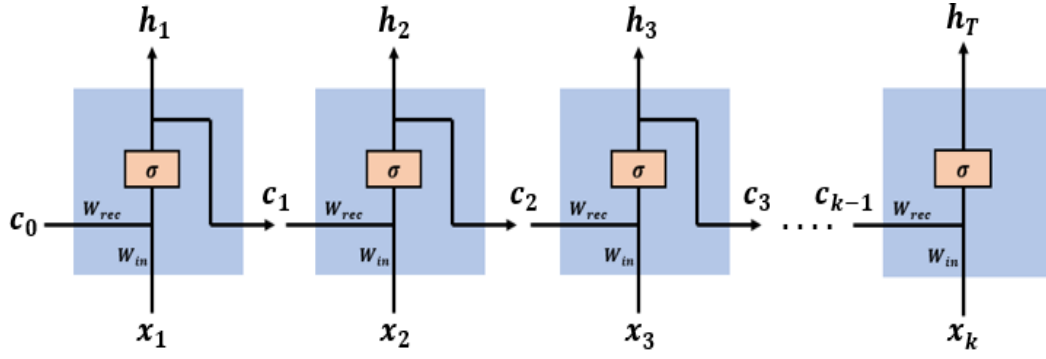
Hình 1.17- Kiến trúc mạng ResNet

### 1.3 Mạng Hồi Quy

Trong khi các mạng nơ-ron tích chập có thể xử lý hiệu quả dữ liệu không gian, các mạng nơ-ron hồi quy được thiết kế để xử lý dữ liệu chuỗi tốt hơn. Các mạng này sử dụng các biến trạng thái để lưu trữ thông tin trong quá khứ, sau đó dựa vào những thông tin này và các đầu vào hiện tại để xác định ra đầu ra hiện tại. Những năm gần đây, RNN đã đem lại những bước đột phá trong nhiều lĩnh vực: nhận dạng giọng nói, mô hình hóa ngôn ngữ, dịch máy, mô tả ảnh,...

#### 1.3.1 Tổng quan

##### a. Kiến trúc mạng hồi quy thông thường



Hình 1.18- Cấu trúc của một mạng hồi quy thông thường

- Mạng hồi quy có một chuỗi đầu vào là các véc tơ  $[x_1, x_2, \dots, x_k]$ . Tại thời điểm  $t$  mạng có véc tơ đầu vào  $x_t$ . Thông tin từ quá khứ và tri thức được mã hóa thành véc-tơ trạng thái  $[c_1, c_2, \dots, c_k]$ . Véc tơ đầu vào  $x_1$  và véc-tơ trạng thái  $c_{t-1}$  được kết hợp lại với nhau tạo thành véc-tơ đầu vào hoàn chỉnh tại thời điểm  $t$ , đó là  $[c_{t-1}, x_t]$
- Mạng hồi quy có 2 ma trận trọng số:  $W_{rec}$ ,  $W_{in}$  tương ứng với  $c_{t-1}$ ,  $x_t$ .
- Hàm **sigmoid** thường được sử dụng như là hàm phi tuyến của tầng ẩn.

### b. Lan truyền ngược theo thời gian trong RNNs

Sau khi mạng hồi quy dự đoán ra véc tơ  $h_k$ , ta tính giá trị lỗi của dự đoán  $E_k$  và sử dụng giải thuật lan truyền ngược để tính đạo hàm:

$$\triangleright \frac{\partial E}{\partial W} = \sum_{t=1}^T \frac{\partial E_t}{\partial W}$$

Ta có đạo hàm lỗi tại thời điểm  $k$ :

$$\begin{aligned} \triangleright \frac{\partial E_k}{\partial W} &= \frac{\partial E_k}{\partial h_k} \frac{\partial h_k}{\partial c_k} \dots \frac{\partial c_2}{\partial c_1} \frac{\partial c_1}{\partial W} \\ &= \frac{\partial E_k}{\partial h_k} \frac{\partial h_k}{\partial c_k} \left( \prod_{t=2}^k \frac{\partial c_t}{\partial c_{t-1}} \right) \frac{\partial c_1}{\partial W} \quad (1) \end{aligned}$$

Lưu ý rằng:  $c_t = \sigma(W_{rec} * c_{t-1} + W_{in} * x_t)$

$$\frac{\partial c_t}{\partial c_{t-1}} = \sigma'(W_{rec} * c_{t-1} + W_{in} * x_t) * \frac{\partial (W_{rec} * c_{t-1} + W_{in} * x_t)}{\partial c_{t-1}}$$

$$= \sigma'(W_{rec} * c_{t-1} + W_{in} * x_t) * W_{rec} \quad (2)$$

Từ (1) và (2):

$$\frac{\partial E_k}{\partial W} = \frac{\partial E_k}{\partial h_k} \frac{\partial h_k}{\partial c_k} \left( \prod_{t=2}^k \sigma(W_{rec} * c_{t-1} + W_{in} * x_t) * W_{rec} \right) \frac{\partial c_1}{\partial W} \quad (3)$$

### c. Vấn đề phụ thuộc xa

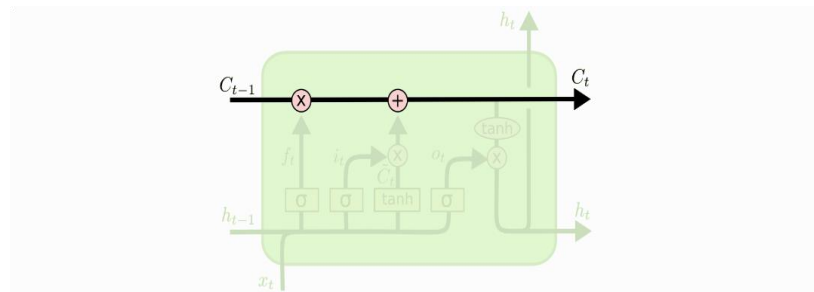
Một điểm nổi bật của RNN chính là ý tưởng kết nối các thông tin phía trước để dự đoán cho hiện tại. Nhưng nhìn vào công thức (3) ta có thể thấy được đạo hàm của hàm lỗi tại bước thứ k có xu hướng bị triệt tiêu với k đủ lớn. Do đó với khoảng cách lớn dần thì RNN bắt đầu không thể nhớ và học được nữa. Đó chính là điểm yếu của mạng hồi quy thông thường. Và để khắc phục nhược điểm này, các mạng LSTM, GRU đã ra đời.

### 1.3.2 Mạng LSTM

Mạng bộ nhớ dài-ngắn (Long Short Term Memory Networks) thường được gọi là LSTM là một dạng đặc biệt của RNN, nó có khả năng học được các phụ thuộc xa. LSTM được giới thiệu bởi Hochreiter & Schmidhuber (1997) và sau đó đã được cải tiến bởi rất nhiều người trong ngành.

#### a. Ý tưởng cốt lõi của LSTM

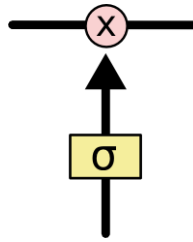
Chìa khóa của LSTM là dùng một đại lượng cell state đóng vai trò như một bộ nhớ cho mô hình. Ở cell state, ta có một đường huyết mạch. Trên con đường huyết mạch này chỉ bao gồm những nút cộng ma trận và phép nhân (element-wise). Khi áp dụng giải thuật lan truyền ngược, đạo hàm của hàm tuyến tính sẽ không bị triệt tiêu. Vì vậy mà thông tin truyền đi xa mà không sợ bị mất mát.



Hình 1.19- Băng chuyền cell state trong LSTM

Thông tin trong cell state có thể được cập nhật: bỏ đi những thông tin không cần thiết và thêm mới những thông tin hữu ích. Nhiệm vụ này được thực hiện một cách chọn lọc bởi các cổng (gate).

Các cổng là nơi sàng lọc thông tin khi thông tin đi qua các cổng, thông tin được kết hợp bởi tầng mạng sigmoid và một phép nhân.



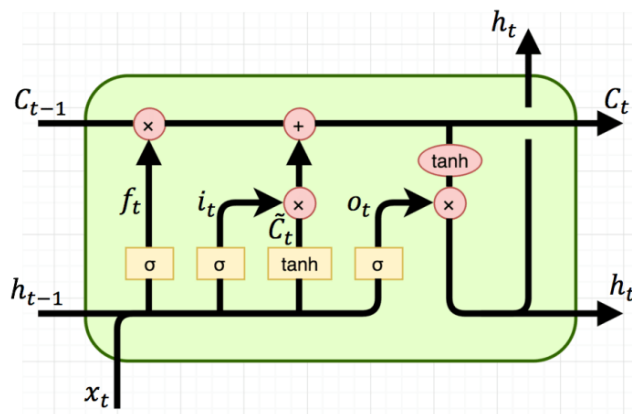
Tầng sigmoid sẽ cho đầu ra là một số trong khoảng  $[0,1]$ , mô tả có bao nhiêu thông tin có thể được thông qua. Khi đầu ra là 0 có nghĩa là không cho ra thông tin nào cả, ngược lại khi là 1 thì có nghĩa tất cả các thông tin được cho qua.

Một LSTM gồm có 3 cổng như vậy để duy trì và điều hành bộ nhớ (cell state).

### ***b. Bên trong LSTM***

Ở state thứ  $t$  của mô hình LSTM:

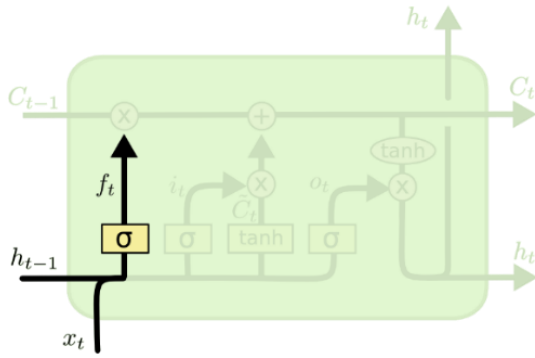
- **Output:**  $c_t, h_t$ , ta gọi  $c$  là cell state,  $h$  là hidden state.
- **Input:**  $c_{t-1}, h_{t-1}, x_t$ . Trong đó  $x_t$  là input ở state thứ  $t$  của model.  $c_{t-1}, h_{t-1}$  là output của layer trước.



Hình 1.20- Cấu trúc của mạng LSTM

Bước đầu tiên của LSTM là quyết định thông tin nào cần bỏ đi từ bộ nhớ (cell state). Quyết định này được đưa ra bởi tầng sigmoid – gọi là cổng quên. Cổng này sẽ lấy đầu vào là  $h_{t-1}, x_t$  rồi đưa ra kết quả là một số trong khoảng  $[0,1]$  được coi là trọng số cho mỗi thông tin trong cell state  $c_{t-1}$ .

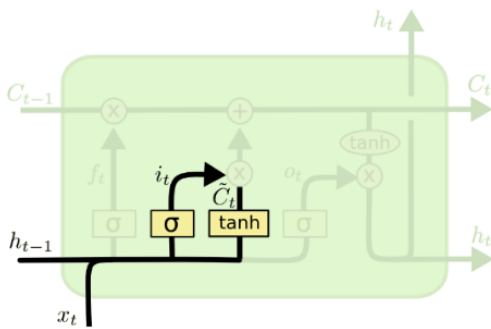




$$f_t = \sigma(W_f \cdot [h_{t-1}, x_t] + b_f)$$

Hình 1.21- Cổng quên

Bước tiếp theo, LSTM quyết định thông tin mới nào sẽ lưu vào bộ nhớ (cell state). Công việc này cần 2 phần. Phần thứ nhất sử dụng một tầng **sigmoid** được gọi là cổng vào để quyết định xem thông tin nào sẽ được cập nhật. Phần thứ hai là một tầng **tanh** tạo ra một véc-tơ cho giá trị  $\tilde{C}_t$  chứa những thông tin cần cập nhật cho trạng thái mới.

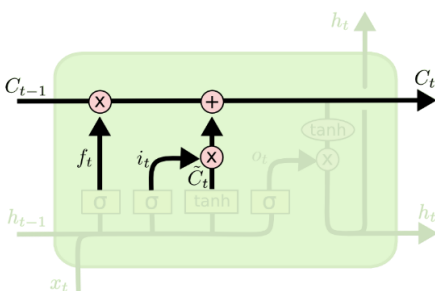


$$i_t = \sigma(W_i \cdot [h_{t-1}, x_t] + b_i)$$

$$\tilde{C}_t = \tanh(W_C \cdot [h_{t-1}, x_t] + b_C)$$

Hình 1.22- Cổng vào

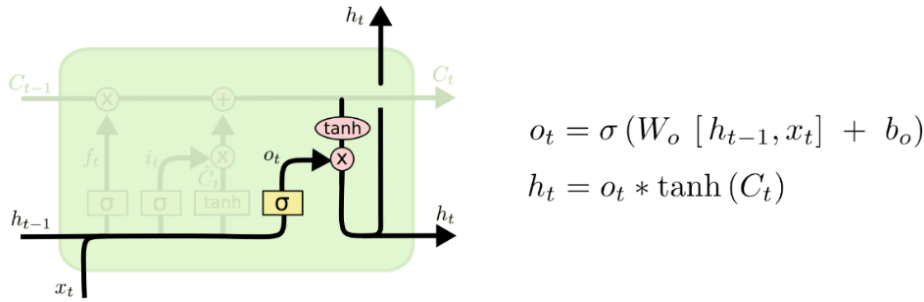
Khi tính toán xong các đại lượng trên, cũng là lúc cập nhật trạng thái (cell state) cũ  $C_{t-1}$  thành trạng thái mới  $C_t$ . Để bỏ đi những thông tin không cần thiết ta nhân trạng thái cũ với  $f_t$ . Sau đó cộng với lượng thông tin mà mô hình muốn thêm vào  $i_t * \tilde{C}_t$ .



$$C_t = f_t * C_{t-1} + i_t * \tilde{C}_t$$

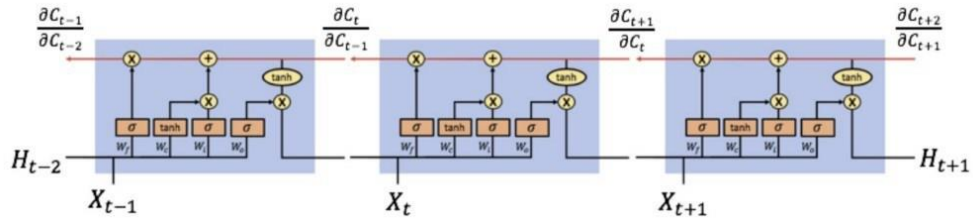
Hình 1.23- Cập nhật trạng thái cell state

Công việc cuối cùng đó là quyết định những thông tin nào sẽ trở thành đầu ra. Giá trị đầu ra sẽ phụ thuộc vào cell state, nhưng sẽ được tiếp tục sàng lọc. Để quyết định xem lấy bao cell state để trở thành đầu ra, mô hình sẽ sử dụng một cổng ra  $o_t$ . Cell state sẽ được đưa qua một hàm **tanh** để có giá trị về khoảng  $[-1,1]$  trước khi nhân với cổng ra để được đầu ra mong muốn.



Hình 1.24- Cổng ra

#### ❖ LSTM chống vanishing gradient



Hình 1.25- LSTM giải quyết vấn đề vanishing gradient

Ta cũng áp dụng thuật toán Lan truyền ngược qua thời gian (**Back Propagation Through Time**) cho LSTM tương tự như RNN.

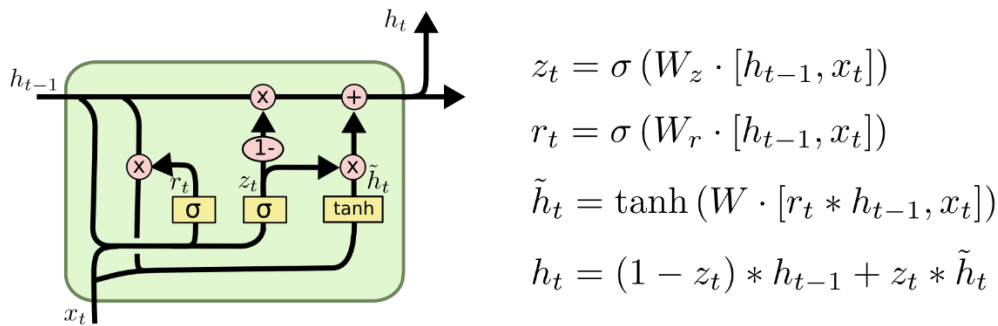
$$\frac{\partial c_t}{\partial c_{t-1}} = f_t . \text{ Do } 0 < f_t < 1 \text{ nên cơ bản thì LSTM vẫn bị triệt tiêu đạo}$$

hàm nhưng bị ít hơn so với RNN. Hơn thế nữa, khi mang thông tin trên cell state thì ít khi cần phải quên giá trị cell cũ, nên  $f_t$  xấp xỉ bằng 1  $\Rightarrow$  Tránh được **vanishing gradient**.

### 1.3.3 Mạng GRU

GRU là phiên bản cải tiến của LSTM, cho ra kết quả tương đương nhưng hội tụ nhanh hơn và thường được sử dụng trong thực tế.

GRU không dùng “cell state” riêng biệt, mà ghép chung với hidden state và kết hợp 2 cổng “forget” và cổng “out put” thành cổng “update”.



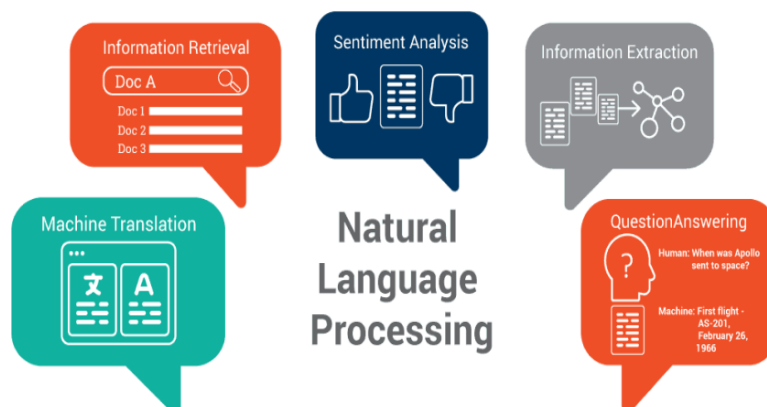
Hình 1.26- Cấu trúc của mạng GRU

- Rest Gate: Quyết định thông tin trạng thái ẩn đằng trước nào là cần thiết để giữ lại để phục vụ cho việc cập nhật trạng thái ẩn tiếp theo.
- Update Gate: Quyết định trạng thái nào cần update.

## 1.4 Xử lý ngôn ngữ tự nhiên

### 1.4.1 Tổng quan về xử lý ngôn ngữ tự nhiên

Xử lý ngôn ngữ tự nhiên là một nhánh trong trí tuệ nhân tạo liên quan đến sự tương tác giữa máy tính và ngôn ngữ của con người. Mục đích của xử lý ngôn ngữ tự nhiên là giúp cho máy tính có khả năng đọc, hiểu và rút ra ý nghĩa từ ngôn ngữ của con người.



Hình 1.27- Các lĩnh vực trong xử lý ngôn ngữ tự nhiên

Một số ứng dụng chính của xử lý ngôn ngữ tự nhiên đó là:

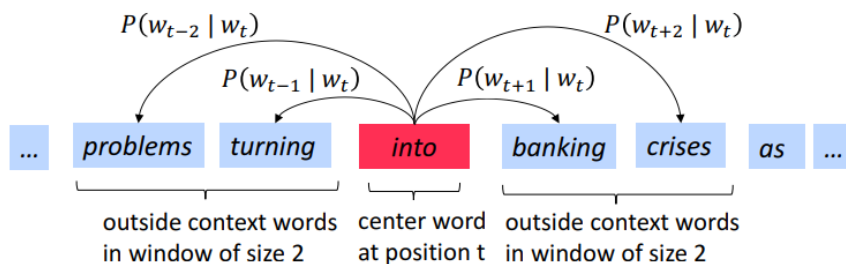
- ❖ Nhận dạng giọng nói
- ❖ Khai phá văn bản
- ❖ Gia sư ngôn ngữ
- ❖ Dịch máy

### 1.4.2 Biểu diễn từ và văn bản

Có rất nhiều kỹ thuật biểu diễn từ trong xử lý ngôn ngữ tự nhiên: WordNet, One-hot, Bag-of-words,.. Tuy nhiên, các kỹ thuật trên đều có những nhược điểm nhất định. Và kỹ thuật word2vec ra đời để khắc phục các nhược điểm của kỹ thuật trước.

#### a. Ý tưởng

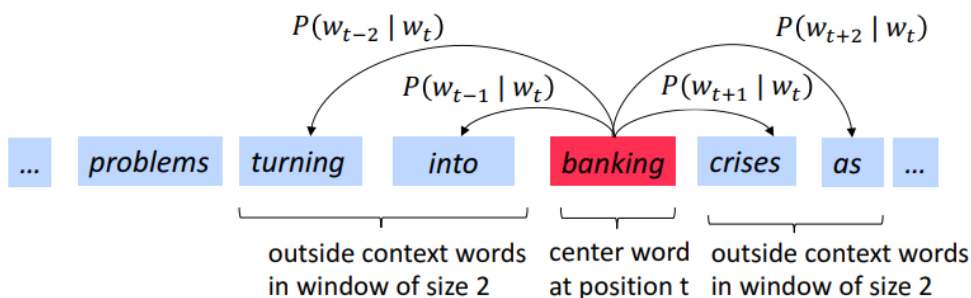
- Sử dụng một tập lớn nhiều văn bản (corpus)
- Mỗi từ trong tập từ vựng cố định được biểu diễn bằng một véc-tơ
- Duyệt từng vị trí  $t$  trong văn bản, mỗi vị trí chứa từ trung tâm  $c$  và các từ ngữ cảnh bên ngoài  $o$
- Sử dụng độ tương đồng của các véc-tơ biểu diễn  $c$  và  $o$  để tính xác suất xuất hiện  $o$  khi có  $c$  (hoặc ngược lại)
- Tinh chỉnh véc-tơ biểu diễn từ để cực đại hóa xác suất này.



Hình 1.28- Ví dụ minh họa về Word2Vec

Như hình trên, các từ ngữ cảnh  $o$  là các từ problems, turning, banking, crises. Từ trung tâm  $c$  là từ into.

Trượt cửa sổ này cho hết câu, ta được các cặp từ trung tâm và ngữ cảnh, đây sẽ là dữ liệu huấn luyện cho thuật toán.



### b. Hàm đánh giá

Giả sử câu có độ dài  $T$ , với mỗi vị trí  $t=1, \dots, T$ , dự đoán ngữ cảnh của từ trong cửa sổ trượt có kích thước  $m$  khi đã biết từ trung tâm  $w_t$ .

Likelihood =  $L(\theta) = \prod_{t=1}^T \prod_{\substack{-m \leq j \leq m \\ j \neq 0}} P(w_{t+j} | w_t; \theta)$

$\theta$  is all variables to be optimized

Hàm lỗi  $J(\theta)$  là đối dấu của giá trị trung bình của logarithm  $L(\theta)$

$$J(\theta) = -\frac{1}{T} \log L(\theta) = -\frac{1}{T} \sum_{t=1}^T \sum_{\substack{-m \leq j \leq m \\ j \neq 0}} \log P(w_{t+j} | w_t; \theta)$$

Việc tối ưu tham số để cực đại hóa dự đoán tương đương với việc cực tiểu hóa hàm lỗi  $J(\theta)$ .

Để cực tiểu hóa hàm  $J(\theta)$ , ta sử dụng 2 véc-tơ cho mỗi từ  $w$ :

- Véc-tơ  $v_w$  khi  $w$  là từ trung tâm
- Véc-tơ  $u_w$  khi  $w$  là một từ trong ngữ cảnh của một từ khác

Sau đó, với mỗi cặp từ trung tâm  $c$  và một ngữ cảnh của nó  $o$ , tính  $P(w_{t+j} | w_t; \theta)$  theo công thức:

$$P(o|c) = \frac{\exp(u_o^T v_c)}{\sum_{w \in V} \exp(u_w^T v_c)}$$

Công thức trên chính là hàm softmax:

$$\text{softmax}(x_i) = \frac{\exp(x_i)}{\sum_{j=1}^n \exp(x_j)} = p_i$$

Trong đó,  $x_i$  là đại lượng đánh giá mức độ tương đồng về mặt ngữ nghĩa giữa từ  $c$  và ngữ cảnh  $o$  và được đo bằng tích vô hướng giữa 2 véc-tơ. Giá trị này càng cao, mức độ tương đồng càng lớn.

## CHƯƠNG 2: MÔ HÌNH SINH MÔ TẢ ẢNH

### 2.1 Tổng quan hệ thống

Sinh mô tả cho ảnh là một vấn đề thách thức trong trí tuệ nhân tạo. Bài toán liên quan đến cả 2 lĩnh vực đó là : Thị giác máy tính và xử lý ngôn ngữ tự nhiên. Mô hình phải làm nhiệm vụ hiểu nội dung của bức ảnh và sau đó chuyển sự hiểu biết đó thành ngôn ngữ tự nhiên.

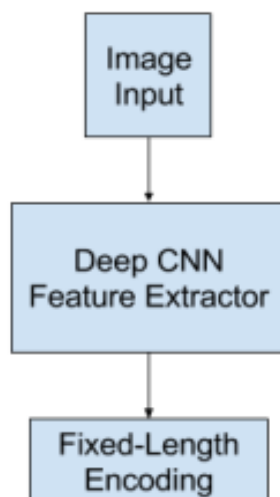
#### 2.1.1 Mô hình sinh mô tả ảnh

Mô hình sinh mô tả ảnh gồm có 2 thành phần chính:

1. Trích xuất đặc trưng
2. Mô hình ngôn ngữ

##### a. Mô hình trích xuất đặc trưng

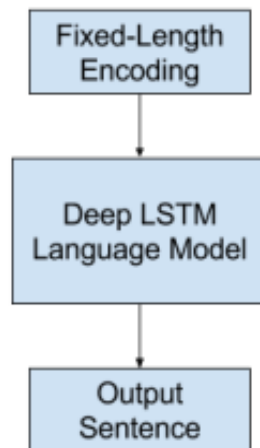
Mô hình trích xuất đặc trưng là một mạng nơ-ron có khả năng trích xuất ra những đặc điểm nổi bật của tấm ảnh, và thường được lưu dưới dạng một véc-tơ có độ dài cố định. Những đặc trưng đã được trích xuất là một biểu diễn của tấm ảnh. Một mạng tích chập thường được sử dụng như là một mô hình trích xuất đặc trưng. Mô hình có thể được huấn luyện trực tiếp với các tấm ảnh trong bộ dữ liệu. Hoặc một mô hình có sẵn (pre-trained) như là Inception, VGG có thể được sử dụng và tinh chỉnh cho phù hợp với bài toán.



Hình 2.1- Trích xuất đặc trưng

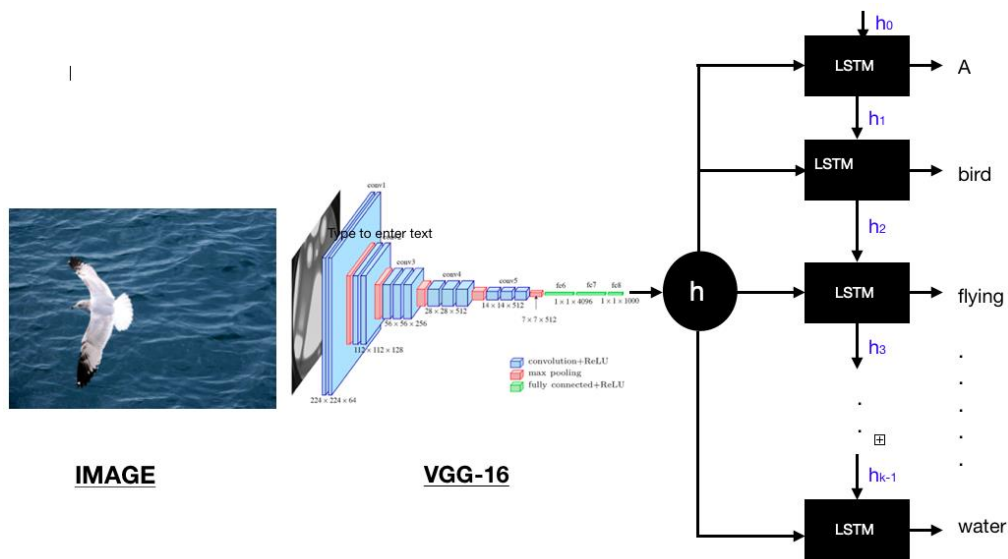
### ***b. Mô hình ngôn ngữ***

Trong việc sinh mô tả cho ảnh, mô hình ngôn ngữ là một mạng nơ-ron mà nhận vào các đặc trưng từ ảnh, có khả năng dự đoán một chuỗi các từ và xây dựng nên mô tả cho ảnh dựa trên những từ đã được sinh ra. Mạng hồi quy được sử dụng phổ biến cho tác vụ này. Tại mỗi time-step đầu ra sẽ là một từ. Mỗi từ mà được tạo ra sẽ được mã hóa bằng phương pháp word embedding và truyền vào mạng như là đầu vào để dự đoán ra từ tiếp theo.



*Hình 2.2- Mô hình ngôn ngữ*

### ***2.1.2 Kiến trúc Encoder-Decoder cho bài toán sinh mô tả ảnh***



Hình 2.3- Mô hình Encoder-Decoder cho bài toán sinh mô tả ảnh

Mô hình giải quyết bài toán sinh mô tả cho ảnh dựa trên kiến trúc **Encoder-Decoder**:

- Encoder CNN: sinh ra “thông tin mã hóa” (encoding) từ bức ảnh
- Decoder LSTM: sinh ra câu đích dựa trên thông tin mã hóa.

## 2.2 Mô hình sinh mô tả ảnh đề xuất

### 2.2.1 Đề xuất cơ chế Chú ý trong bài toán sinh mô tả ảnh

#### a. Vấn đề với kiến trúc Encoder-Decoder thông thường

Hệ thống sinh mô tả cho ảnh thông thường sẽ mã hóa (encode) bức ảnh sử dụng một mô hình có sẵn (pretrained) để sinh ra hidden state **h**. Sau đó hệ thống sẽ giải mã (decode) hidden state này bằng một RNN và sinh ra các từ lần lượt cho đến khi thu được một đoạn mô tả hoàn chỉnh. Mô hình này đã trình bày ở phần trên (Hình 20)

Vấn đề của phương pháp này là khi hệ thống muốn sinh ra từ tiếp theo của mô tả thì từ tiếp theo này thường chỉ mô tả một phần nào đó của tấm hình. Trong khi đó, ta lại sử dụng cả hidden state **h** chỉ để mô tả một phần nhỏ của bức ảnh và việc này không hiệu quả. Vì thế cơ chế Chú ý sinh ra để giải quyết vấn đề này.

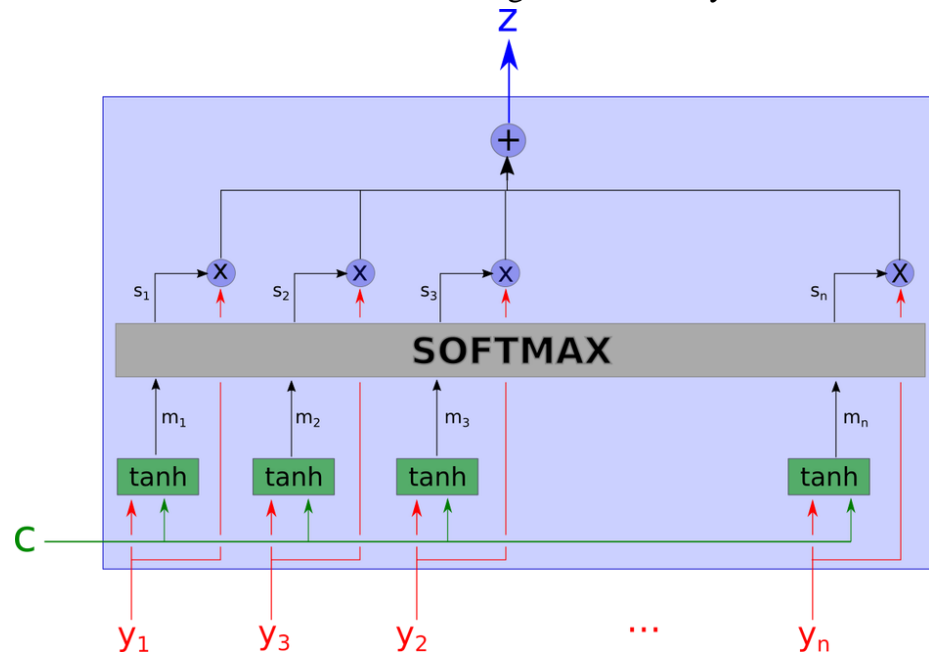


Trong cơ chế chú ý, bức ảnh sẽ chia ra làm nhiều phần và khi mạng RNN dự đoán từ tiếp theo, cơ chế này sẽ giúp RNN tập trung vào những phần ảnh có liên quan để làm nhiệm vụ dự đoán của mình.

### b. Mô hình Attention

Mô hình attention này được lấy ý tưởng từ paper “Neural machine translation by jointly learning to align and translate” của nhóm tác giả Bahdanau, Dzmitry, Bengio, Yoshua vào năm 2014.

Một mô hình Attention sẽ nhận đầu vào là  $n$  phần của tấm ảnh  $y_1, y_2, \dots, y_n$  và một véc-tơ ngữ cảnh  $c$ . Mô hình sẽ trả về một véc-tơ  $z$  được gọi là “tóm tắt” của toàn bộ tấm hình tương ứng với ngữ cảnh  $c$ . Cụ thể hơn nữa mô hình sẽ trả về các trọng số đánh giá mức độ quan trọng của các phần  $y_1, y_2, \dots, y_n$  cho việc dự đoán từ tiếp theo. Chi tiết mô hình được minh họa bằng hình dưới đây:



Hình 2.4– Mô hình Attention

Với đầu vào là véc-tơ ngữ cảnh  $c$  và các phần của bức ảnh  $y_i$ , mô hình sẽ tính toán  $m_1, m_2, \dots, m_n$  là véc-tơ kết hợp của  $c$  và  $y_i$ .

$$\square m_i = \tanh(W_{cm} * c + W_{ym} * y_i)$$

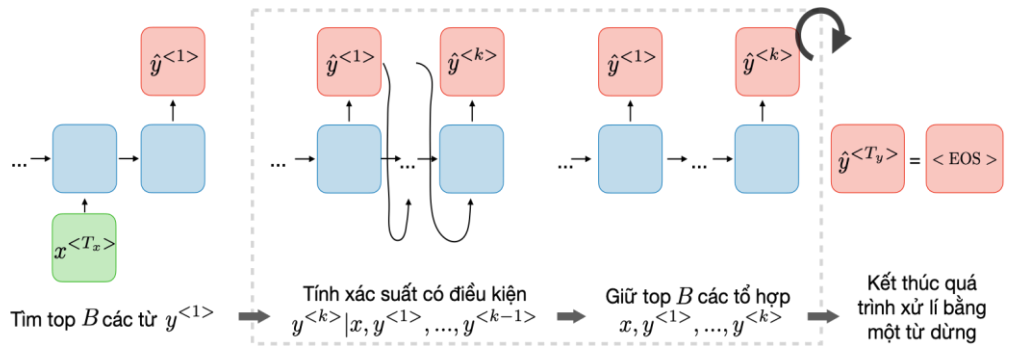
Sau đó, các trọng số tương ứng với mỗi  $y_i$  sẽ được tính thông qua hàm Softmax. Giá trị của các trọng số này sẽ thay đổi một cách tương ứng khi ngữ cảnh  $c$  thay đổi, điều này phù hợp với việc ta dịch chuyển sự chú ý đến các phần tiếp theo của ảnh. Mô hình này có thể áp dụng vào bất kì hệ thống nào với giá trị đạo hàm sẽ được học qua giải thuật lan truyền ngược.

### 2.2.2 Đề xuất tìm kiếm chùm

Trong mô hình sinh mô tả ảnh, bộ giải mã (decoder) làm nhiệm vụ sinh ra mô tả đích. Tại thời điểm  $t$ , bộ giải mã cần phải quyết định từ nào sẽ là từ thứ  $t$  trong mô tả. Và ta mong muốn câu đích sinh ra sẽ cực đại hóa xác suất hậu nghiệm.

Trước đây, giải thuật tham lam được áp dụng cho bài toán sinh mô tả ảnh. Bộ giải mã sẽ sinh ra câu đích bằng cách lấy argmax từng bước. Nhược điểm của phương pháp này là nếu bộ giải mã sai ở bước nào thì các bước tiếp theo sẽ sai toàn bộ, không có cách nào sửa chữa.

Để giải quyết vấn đề này thuật toán tìm kiếm chùm đã được áp dụng, với ý tưởng là tại mỗi bước giải mã ta duy trì  $k$  phương án bộ phận có xác suất xảy ra cao nhất (gọi là các giả thuyết).



Hình 2.5 – Giải thuật tìm kiếm chùm trong dịch máy

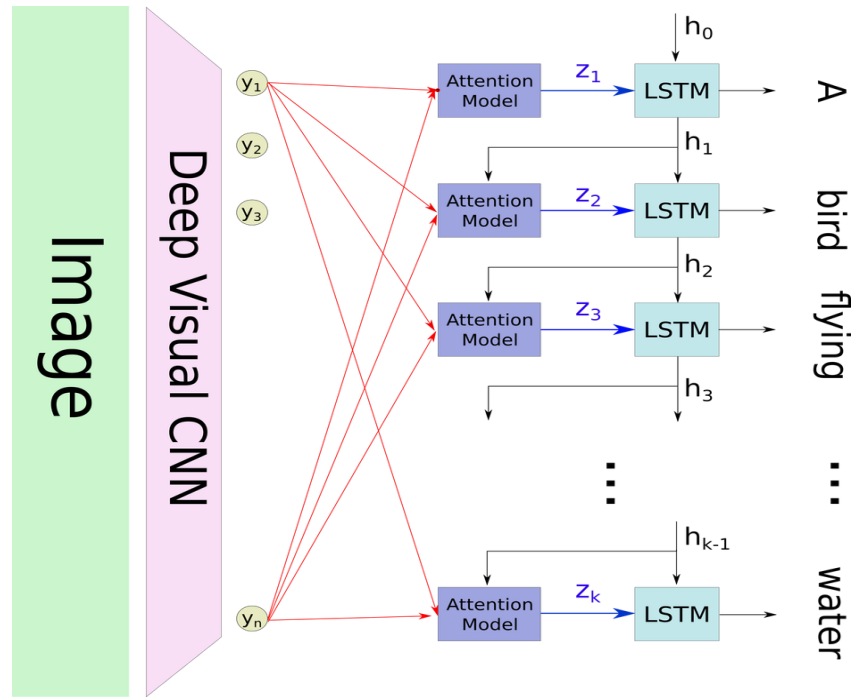
Một giả thuyết  $y_1, y_2, \dots, y_t$  có điểm bằng log của giá trị xác suất của nó:

$$\text{score}(y_1, \dots, y_t) = \log P_{\text{LM}}(y_1, \dots, y_t | x) = \sum_{i=1}^t \log P_{\text{LM}}(y_i | y_1, \dots, y_{i-1}, x)$$

- Ở đây, tất cả các score đều âm, điểm càng cao càng tốt. Tại mỗi bước ta sẽ giữ lại  $k$  giả thuyết có điểm cao nhất tại mỗi bước.

Tuy rằng thuật toán tìm kiếm chùm không đảm bảo lời giải tối ưu. Nhưng được kì vọng rằng lời giải đó đủ tốt và nó đã được chứng minh rằng hiệu quả hơn rất nhiều so với thuật toán tham lam.

### 2.2.3 Chi tiết mô hình



Hình 2.6- Mô hình sinh mô tả ảnh với cơ chế Attention

#### a. Encoder: CNN

Mô hình nhận đầu vào một ảnh và sinh ra một mô tả y được mã hóa như là một chuỗi gồm k từ:

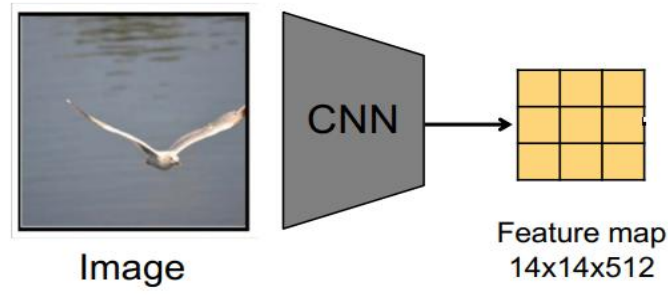
$$y = \{y_1, \dots, y_C\}, y_i \in \mathbb{R}^K$$

Trong đó K là kích thước của tập từ vựng và C là chiều dài của mô tả.

Chúng ta sử dụng một mạng nơ-ron tích chập để trích xuất ra các véc-tơ đặc trưng (feature vectors) mà có tên gọi là các véc-tơ chú thích (annotation vectors). Mạng tích chập tạo ra L véc-tơ, mỗi véc-tơ có độ dài D tương ứng với một phần của bức ảnh.

$$a = \{a_1, \dots, a_L\}, a_i \in \mathbb{R}^D$$

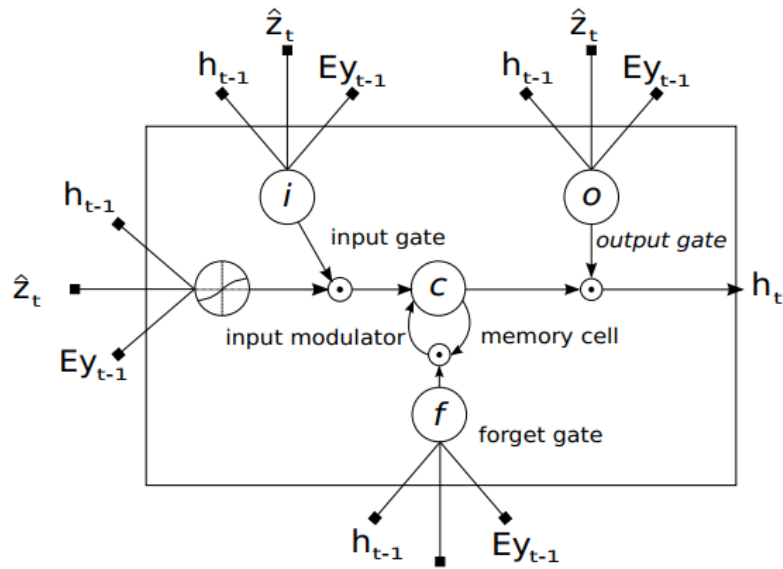
Để có được sự tương ứng giữa các véc-tơ đặc trưng và các phần của tấm ảnh 2D, chúng ta trích xuất đặc trưng từ tầng tích chập thấp hơn thay vì sử dụng cả lớp kết nối đầy đủ (fully connected layer). Điều này giúp cho Decoder tập chung có chọn lọc vào những phần cụ thể của bức ảnh bằng cách lựa chọn ra một tập con của tất cả các véc-tơ đặc trưng.



Hình 2.7- Minh họa Encoder CNN

### b. Decoder: LSTM

Mô hình sử dụng một mạng LSTM sinh ra mô tả bằng cách dự đoán ra một từ tại mỗi bước dựa trên 1 véc-tơ ngữ cảnh (context vector), hidden state trước đó và từ được dự đoán ở time step trước.



Hình 2.8- Mạng LSTM với cơ chế chú ý

Sử dụng  $T_{s,t} : R^s \rightarrow R^t$  để kí hiệu một biến đổi affine với các tham số đã được học:

$$\begin{pmatrix} \mathbf{i}_t \\ \mathbf{f}_t \\ \mathbf{o}_t \\ \mathbf{g}_t \end{pmatrix} = \begin{pmatrix} \sigma \\ \sigma \\ \sigma \\ \tanh \end{pmatrix} T_{D+m+n,n} \begin{pmatrix} \mathbf{E}\mathbf{y}_{t-1} \\ \mathbf{h}_{t-1} \\ \hat{\mathbf{z}}_t \end{pmatrix} \quad (1)$$

$$\mathbf{c}_t = \mathbf{f}_t \odot \mathbf{c}_{t-1} + \mathbf{i}_t \odot \mathbf{g}_t \quad (2)$$

$$\mathbf{h}_t = \mathbf{o}_t \odot \tanh(\mathbf{c}_t). \quad (3)$$

Trong đó,  $i_t$ ,  $f_t$ ,  $c_t$ ,  $o_t$ ,  $h_t$  lần lượt là cổng vào, cổng quên, cell nhớ, cổng ra và hidden state của LSTM. Véc-tơ  $\hat{\mathbf{z}}_t \in R^D$  là véc-tơ ngữ cảnh.  $\mathbf{E} \in R^{m \times K}$  là ma trận embedding, m và n lần lượt là số chiều của word vector và kích thước của mạng LSTM.

Véc-tơ ngữ cảnh  $\hat{\mathbf{z}}_t$  là một biểu diễn động của một phần bức ảnh có liên quan tại thời điểm t. Chúng ta xác định hàm  $\emptyset$  để tính toán  $\hat{\mathbf{z}}_t$  từ véc-tơ chú thích (annotation)  $\mathbf{a}_i$ , với  $i = 1, \dots, L$  tương ứng với các đặc trưng được trích xuất tại các vị trí khác nhau của tấm hình. Mỗi vị trí  $i$ , hàm sẽ tạo ra một trọng số dương  $\alpha_i$  mà có thể được hiểu là xác suất vị trí thứ  $i$  là nơi mà mô hình cần tập trung vào để sinh ra từ tiếp theo hoặc là tầm quan trọng của vị trí  $i$  trong việc kết hợp các  $\mathbf{a}_i$  lại với nhau. Trọng số  $\alpha_i$  của mỗi véc-tơ chú thích  $\mathbf{a}_i$  được tính bởi mô hình attention  $f_{att}$ . Mô hình attention dùng một mạng perceptron đa lớp (Multi-layer perceptron) dựa trên hidden state trước đó  $\mathbf{h}_{t-1}$ . Do đó, mô hình “nhìn” vào nơi nào của bức ảnh để dự đoán từ tiếp theo hoàn toàn phụ thuộc vào chuỗi các từ đã được sinh ra trước đó.

$$e_{ti} = f_{att}(\mathbf{a}_i, \mathbf{h}_{t-1}) \quad (4)$$

$$\alpha_{ti} = \frac{\exp(e_{ti})}{\sum_{k=1}^L \exp(e_{tk})}. \quad (5)$$

Khi mà trọng số đã được tính toán, thì véc-tơ ngữ cảnh  $\hat{\mathbf{z}}_t$  tính toán bằng công thức:

$$\hat{\mathbf{z}}_t = \phi(\{\mathbf{a}_i\}, \{\alpha_i\})$$

Trong đó  $\emptyset$  nhận đầu vào là tập véc-tơ chú thích  $\mathbf{a}_i$  và tập các trọng số tương ứng của chúng, đầu ra là một véc-tơ.

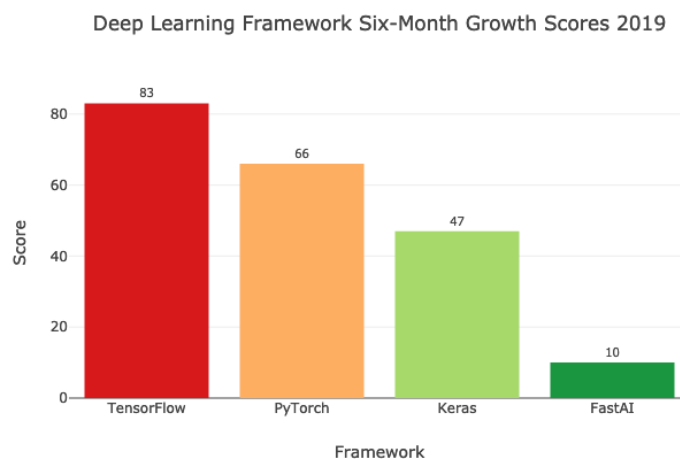
## CHƯƠNG 3: CÀI ĐẶT CHI TIẾT VÀ ĐÁNH GIÁ

### 3.1 Môi trường cài đặt mô hình và công cụ sử dụng

#### 3.1.1 Môi trường cài đặt mô hình

Mô hình được cài đặt bằng ngôn ngữ Python và sử dụng thư viện chuyên dùng cho Deep learning là Tensorflow 2.0

Tensorflow là thư viện mã nguồn mở dùng cho tính toán số học, đặc biệt phù hợp cho những bài toán học máy có quy mô lớn. Nó được phát triển bởi nhóm Google Brain. Tensorflow là thư viện mã nguồn mở vào tháng 11 năm 2015 và hiện tại nó là thư viện học sâu phổ biến nhất. Vô số dự án sử dụng Tensorflow cho các loại tác vụ học máy như là phân loại ảnh, xử lý ngôn ngữ tự nhiên, hệ trợ giúp quyết định,...



Hình 3.1- Điểm tăng trưởng của các thư viện deep learning trong năm 2019

#### 3.1.2 Công cụ sử dụng

Mô hình được huấn luyện trên Google colab. Google Colab là một dịch vụ đám mây miễn phí, có hỗ trợ GPU (Tesla K80) và TPU (TPUv2). Google Colab là một dịch vụ miễn phí tuyệt vời của google nếu người dùng không có một máy tính cấu hình cao để lập trình, biên dịch Python với các thư viện của học sâu. Hiện Google Colab có sẵn các thư viện phổ biến hỗ trợ cho việc nghiên cứu về Trí tuệ nhân tạo như: PyTorch, Tensorflow, Keras và OpenCV.

### 3.2 Thu thập dữ liệu

Bộ dữ liệu dùng cho bài toán này là Flick8k và Flick30k. Trong đó, bộ dữ liệu Flick 8k chứa 8000 ảnh, bộ dữ liệu Flick30k có 30000 ảnh và mỗi ảnh có 5 mô tả. Những hình ảnh trong Flick8k được chia thành các phần sau:

- Tập huấn luyện: 6000 ảnh
- Tập validate: 1000 ảnh
- Tập test: 1000 ảnh



a wet black dog is carrying a green toy through the grass .  
a dog in grass with a blue item in his mouth .  
a black dog has a blue toy in its mouth .  
a black dog carrying something through the grass .  
a black dog carries a green toy in his mouth as he walks through the grass .



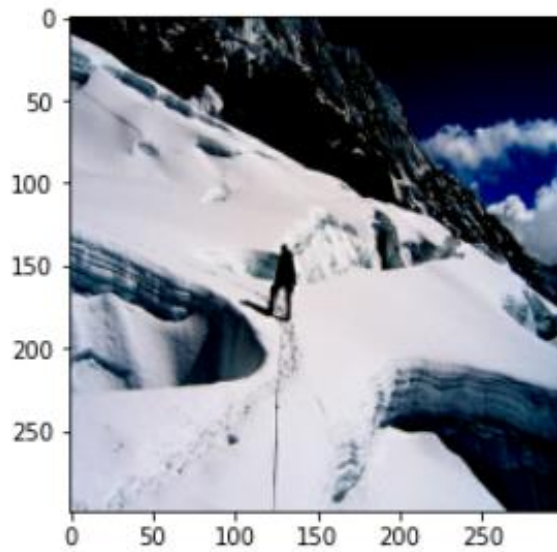
man and child in yellow kayak  
a man and young boy ride in a yellow kayak .  
a man and child kayak through gentle waters .  
a man and a little boy in blue life jackets are rowing a yellow canoe .  
a man and a baby are in a yellow kayak on water .

*Hình 3.2- Dữ liệu dùng để huấn luyện mô hình*

### 3.3 Tiền xử lý dữ liệu ảnh

- ❖ Chúng ta cần chuyển mỗi ảnh trong tập dữ liệu về kích thước cố định (299, 299, 3) để đưa vào mạng nơ-ron để trích xuất đặc trưng.

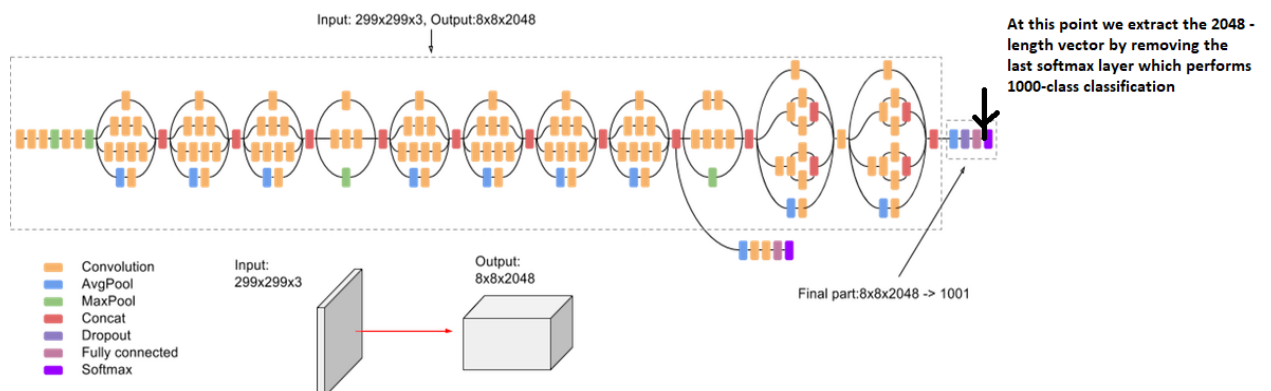
Shape after resize : (299, 299, 3)  
 <matplotlib.image.AxesImage at 0x7f28460badd8>



Hình 3.3- Ảnh sau khi đã resize về kích thước (299, 299,3)

❖ Định nghĩa mô hình pre-trained cho ảnh:

- Mô hình được sử dụng ở đây là Inception v3. Inception v3 là một mạng tích chập được nghiên cứu bởi google. Mô hình này đã được huấn luyện trên tập dữ liệu cho bài toán phân loại ảnh với 1000 class khác nhau và đạt độ chính xác lên đến 93.9 %. Sử dụng mô hình tốt như Inception v3 giúp cho mô hình sinh mô tả ảnh có khả năng nhận diện được các đối tượng khác nhau trong ảnh tốt hơn, cho phép tạo ra các mô tả chi tiết và chính xác hơn.



Hình 3.4- Mô hình trích xuất đặc trưng (Inception v3)



- Vì mục đích của bài toán không phải là phân loại ảnh nên tầng liên kết đầy đủ, với hàm kích hoạt là softmax được bỏ đi và đầu ra sẽ là các véc-tơ đặc trưng có độ dài 2048

```
[[0.7555921 0.10535137 0. ... 0. 1.1878283 0.05133845]
 [2.192422 0. 0.5397214 ... 0. 0.98265326 0. ]
 [3.5808098 0. 0. ... 0. 1.1083212 0.28665215]
 ...
 [0. 0. 0. ... 0. 0. 0.40023074]
 [0. 0. 0. ... 0. 0. 0.4033794 ]
 [0.11506022 0. 0. ... 0. 0. 0.716787 ]]
Shape : (64, 2048)
```

### 3.4 Tiền xử lí mô tả

#### ❖ Làm sạch dữ liệu:

- Khi xử lí dữ liệu dạng văn bản, các bước cơ bản cần thực hiện đó là:
  - ✓ Chuyển chữ hoa thành chữ thường, “Hello” thành “hello”
  - ✓ Bỏ các kí tự đặc biệt “%”, “\$”, “#”
  - ✓ Loại bỏ các chữ có số như hey 199
- Loại bỏ đi những từ chứa ít thông tin như a, an,... Để giúp mô hình hoạt động tốt hơn.

```
['A black dog and a spotted dog are fighting',
 'A black dog and a tri-colored dog playing with each other on the road .',
 'A black dog and a white dog with brown spots are staring at each other in the street .',
 'Two dogs of different breeds looking at each other on the road .',
 'Two dogs on pavement moving toward each other .']
```

Hình 3.5- Dữ liệu trước khi được làm sạch

```
['black dog and spotted dog are fighting',
 'black dog and tricolored dog playing with each other on the road',
 'black dog and white dog with brown spots are staring at each other in the street',
 'two dogs of different breeds looking at each other on the road',
 'two dogs on pavement moving toward each other']
```

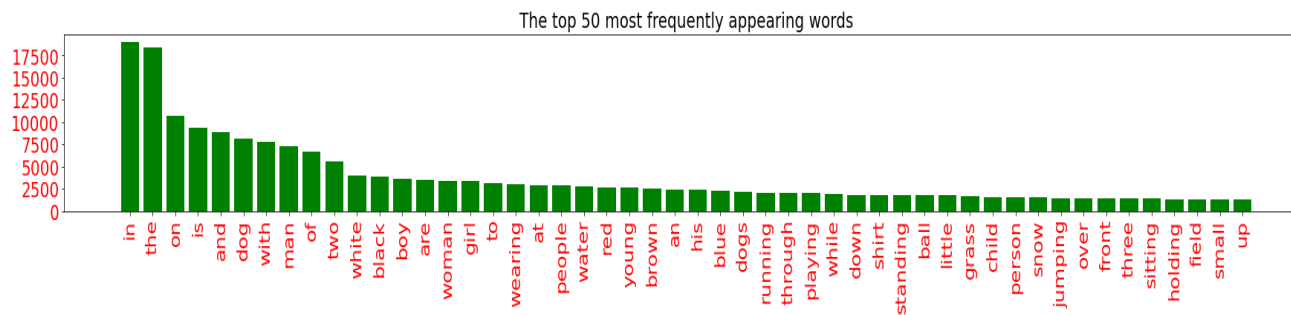
Hình 3.6- Dữ liệu sau khi được làm sạch

- ❖ Thêm 2 thẻ <start> và <end> vào mỗi mô tả để cho mô hình biết được bắt đầu và kết thúc của một mô tả

```
train_captions[:3]
```

```
['<start> man jumps bicycle down porch stairway too close to the fence <end>',
 '<start> man with glasses and tie plays Jenga with woman in green shirt <end>',
 '<start> Three kids and an adult playing water volleyball in pool <end>']
```

- ❖ Tạo từ điển gồm các từ đại diện trong toàn bộ mô tả
  - Có 2756 từ khác nhau trong số 40000 caption. Tuy nhiên có một số từ chỉ xuất hiện 1 vài lần, nó giống như làm nhiễu và không tốt cho việc học dự toán từ của model. Do đó ta chỉ giữ lại 12000 từ xuất hiện nhiều nhất trong tất cả mô tả.




Hình 3.7- Top 50 từ có tần suất xuất hiện nhiều nhất trong dataset


### 3.5 Xây dựng mô hình

- ❖ Định nghĩa mô hình Encoder (Inception v3)

```
class CNN_Encoder(tf.keras.Model):
    # Since you have already extracted the features and dumped it using pickle
    # This encoder passes those features through a Fully connected layer
    def __init__(self, embedding_dim):
        super(CNN_Encoder, self).__init__()
        # shape after fc == (batch_size, 64, embedding_dim)
        self.fc = tf.keras.layers.Dense(embedding_dim)

    def call(self, x):
        x = self.fc(x)
        x = tf.nn.relu(x)
        return x
```

 `encoder.summary()`

 Model: "cnn\_\_encoder"

Layer (type)	Output Shape	Param #
dense (Dense)	multiple	524544

=====  
 Total params: 524,544  
 Trainable params: 524,544  
 Non-trainable params: 0

Hình 3.8- Tham số của encoder

❖ Định nghĩa mô hình Decoder với cơ chế Attention

- Mô hình attention được dựa trên Paper Bahdanau

Model: "bahdanau\_attention"

Layer (type)	Output Shape	Param #
dense_3 (Dense)	multiple	131584
dense_4 (Dense)	multiple	262656
dense_5 (Dense)	multiple	513

=====  
 Total params: 394,753  
 Trainable params: 394,753  
 Non-trainable params: 0

Hình 3.9- Tham số trong mô hình Attention

- Mạng hồi quy trong Decoder, mô hình sử dụng GRU thay cho LSTM để rút ngắn thời gian huấn luyện nhưng vẫn có kết quả tương đương.

Model: "rnn\_decoder"

Layer (type)	Output Shape	Param #
embedding (Embedding)	multiple	1280256
gru (GRU)	multiple	1575936
dense_1 (Dense)	multiple	262656
dense_2 (Dense)	multiple	2565513
bahdanau_attention (Bahdanau multiple		394753
dropout (Dropout)	multiple	0
batch_normalization_94 (Batc multiple		2048
Total params: 6,081,162		
Trainable params: 6,080,138		
Non-trainable params: 1,024		

Hình 3.10- Tham số trong mô hình Decoder (RNN) với cơ chế Attention

### 3.6 Huấn luyện mô hình

❖ Thông số mô hình:

```
# Parameters of system's configuration
top_k = 5000
BATCH_SIZE = 64
BUFFER_SIZE = 1000
embedding_dim = 256
units = 512
vocab_size = top_k + 1
num_steps = len(img_name_train) // BATCH_SIZE
# Shape of the vector extracted from InceptionV3 is (49, 512)
# These two variables represent that vector shape
features_shape = 2048
attention_features_shape = 64
```

Hình 3.11- Thông số mô hình

Sau nhiều lần tinh chỉnh thì Mô hình cho kết quả tốt nhất với BATCH\_SIZE = 64, kích thước từ điển vocab\_size = 12000, số chiều

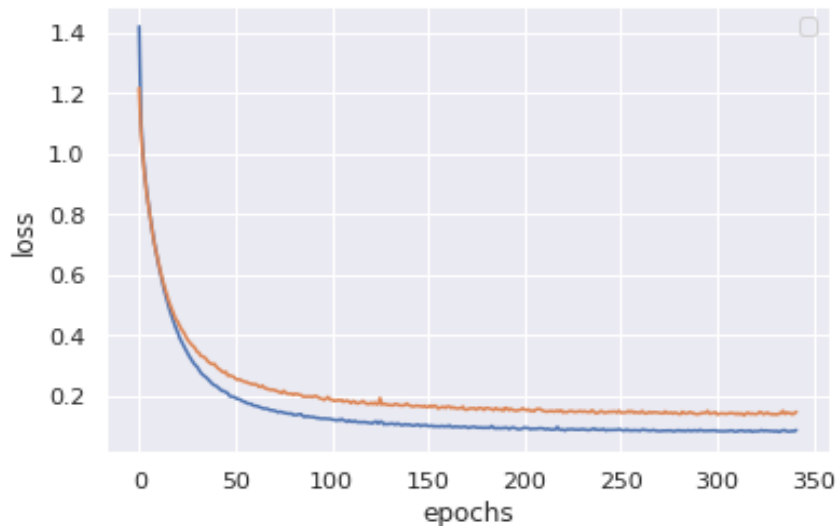
của véc-tơ word embedding  $\text{embedding\_dim} = 256$ , số chiều của hidden state trong mạng GRU  $\text{units} = 512$ .

❖ Hàm mất mát:

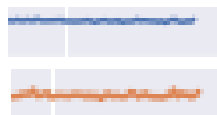
- Loss function  $= \frac{1}{T} \sum_{t=1}^T (J_t)$

Trong đó  $J_t$  là negative log probability của từ đúng trong mô tả tại thời điểm  $t$ .

❖ Trong quá trình train, thuật toán tối ưu sử dụng là Adam với  $\text{beta1} = 0.9$ ,  $\text{beta2} = 0.999$  và  $\text{learning\_rate} = 1e-3$ . Sau 360 epochs, mô hình hoàn thành việc huấn luyện:



Hình 3.12- Giá trị của hàm loss function qua thời gian trên tập train và validate



: Train loss

: Valid loss

### 3.7 Kết quả thực nghiệm

#### 3.7.1 Độ đo

Cũng như trong mô hình dịch máy, để đánh giá một mô hình sinh mô tả cho ảnh có hoạt động tốt hay không, ta thường sử dụng BLEU Score. BLEU Score là một

độ đo hay một hệ số điểm khi so sánh một bản dịch máy (candidate translation) với một hay nhiều bản dịch tham khảo (reference translation).

Cách tính BLEU là phương pháp đếm số matching n-grams của bản dịch máy và bản dịch tham khảo, kết quả sẽ là số trùng khớp chia cho số từ của candidate. Cách match này không phụ thuộc vào vị trí, do vậy BLEU không yêu cầu trật tự các từ và càng nhiều match thì càng tốt.

Công thức tính BLEU được chia làm 2 phần: modified n-gram precision score và Brevity Penalty.

Một cụm N-gram là 1 dãy con gồm n phần tử liên tiếp nhau của 1 dãy các phần tử cho trước.

Modified n-gram precision score kí hiệu là  $P_n$  là tỉ lệ số n-gram ứng viên xuất hiện trong văn bản tham khảo trên tổng số n-gram của ứng viên.

$$P_n = \frac{\sum_{n\text{-grams} \in \hat{y} \text{ count}_{clip}(n\text{-grams})}{\sum_{n\text{-grams} \in y \text{ count}(n\text{-grams})}, \text{ với } \hat{y} \text{ là câu dịch máy}$$

Để tổng hợp tất cả các modified n-gram precision score, ta sử dụng geometric mean, được định nghĩa như sau:

$$\text{Precision} = \exp\left(\sum_{n=1}^N \frac{\log P_n}{n}\right)$$

Các văn bản không thể có độ dài tùy tiện nên ta sử dụng Brevity Penalty để xử lí những bản dịch có độ dài ngắn:

$$\text{Brevity Penalty} = \begin{cases} 1, & r < c \\ \exp(1 - \frac{c}{r}), & r \geq c \end{cases}$$

trong đó: c là độ dài bản dịch, r là độ dài bản tham khảo.

Cuối cùng điểm, BLEU được tính theo công thức:

$$\text{BLEU} = \text{Precision} * \text{Brevity Penalty}$$

Từ công thức có thể thấy điểm BLEU luôn trong khoảng từ 0 đến 1. Điểm càng cao, thì độ trùng khớp giữa bản dịch máy với bản dịch tham khảo càng lớn. Tuy nhiên BLEU phụ thuộc nhiều vào số lượng, cũng như chất lượng của bản dịch nên đây cũng là một trong những khó khăn khi đánh giá mô hình.

### 3.7.2 Kết quả đánh giá trên bộ dữ liệu Flickr8k

Thông thường trong các bài báo nghiên cứu, để đánh giá chất lượng mô tả cho ảnh người ta hay sử dụng cumulative BLEU-1, BLEU-2, BLEU-3, BLEU-4 với các trọng số của n-grams như sau:

	1-gram	2-gram	3-gram	4-gram
BLEU-1	1	0	0	0
BLEU-2	0.5	0.5	0	0
BLEU-3	0.33	0.33	0.33	0
BLEU-4	0.25	0.25	0.25	0.25

*Bảng 3.1- Trọng số của n-grams ứng với các loại điểm BLEU khác nhau*

Sau khi tính toán, điểm BLEU trung bình trên tập flick 8k test có kết quả:

BLEU-1	53
BLEU-2	37.2
BLEU-3	26.4
BLEU-4	16.7

*Bảng 3.2- Số điểm BLEU trên tập dữ liệu Flick8k test*

So sánh với kết quả của nhóm nghiên cứu của đại học Stanford trong paper “Show, Attend and Tell: Neural Image Caption Generation with Visual Attention”: số điểm BLEU của mô hình có số điểm thấp hơn một chút.

BLEU-1	67
BLEU-2	44.8
BLEU-3	29.9
BLEU-4	19.5

*Bảng 3.3- Điểm BLEU trên tập dữ liệu Flick8k test do đại học stanford thực hiện*

### **3.7.3 Một vài kết quả từ bộ dữ liệu**

Real Caption: an adult holds small child who sits on table in mall food court  
Prediction Caption: baby is held by an adult and is sitting on table at coffee  
time took to Predict: 5 sec



*Hình 3.13- Một vài kết quả từ bộ dữ liệu (1)*

Real Caption: two girls with ponytails ride an amusement park ride  
Prediction Caption: two girls on ride at an amusement park



*Hình 3.14- Một vài kết quả thu được từ bộ dữ liệu (2)*



Real Caption: the boys are playing with legos

Prediction Caption: two boys play with legos



*Hình 3.15- Một vài kết quả thu được từ bộ dữ liệu (3)*

Real Caption: dog jumping for frisbee

Prediction Caption: black and white dog catching frisbees in midair  
time took to Predict: 4 sec



*Hình 3.16- Một vài kết quả thu được từ bộ dữ liệu (4)*

Real Caption: two older women walking down the busy street with shopping bags  
Prediction Caption: group of people are walking down busy city street  
time took to Predict: 3 sec



*Hình 3.17- Một vài kết quả thu được từ bộ dữ liệu (5)*

### 3.8 Đánh giá mô hình

Mô hình có 2 ưu điểm đó là: cơ chế chú ý (Attention) và giải thuật tìm kiếm chùm ( Beam Search). Nhờ vào các ưu điểm này mà mô hình hoạt động khá tốt, đưa ra được những mô tả phù hợp với bức ảnh, thậm chí với những bức ảnh phức tạp. Tuy nhiên, cũng giống như bao mô hình học máy khác, khi chúng ta dùng tập dữ liệu thử nghiệm không có liên quan về mặt ngữ nghĩa với tập dữ liệu huấn luyện, mô hình sẽ sinh ra những mô tả không tốt. Ví dụ: nếu chúng ta huấn luyện mô hình về hình ảnh mèo, chó,... thì không nên thử nghiệm mô hình về những hình ảnh máy bay, thác nước,... vì mô hình chưa từng nhìn thấy chúng bao giờ.

## CHƯƠNG 4: KẾT LUẬN VÀ ĐỊNH HƯỚNG PHÁT TRIỂN

### 4.1 Kết luận

Đồ án này trình bày về quá trình tìm hiểu, nghiên cứu cơ sở lý thuyết của mạng nơ-ron tích chập, mạng hồi quy, mô hình xử lý ngôn ngữ tự nhiên và các kỹ thuật học sâu trong học máy. Qua đó áp dụng những kiến thức này vào bài toán sinh mô tả cho ảnh. Bài toán này là sự kết hợp của 2 lĩnh vực xử lý ảnh và xử lý ngôn ngữ tự nhiên. Đặc biệt, mô hình còn được cải thiện nhờ vào cơ chế Attention. Trong hội thảo về Deep Learning ICLR 2019, người ta đã thống kê rằng số bài báo về Attention có xu hướng tăng, còn bài báo liên quan đến mạng hồi quy giảm xuống và được thay thế bởi mô hình Attention. Mô hình Attention đang là xu thế mới, vì nó giải quyết được vấn đề triệt tiêu đạo hàm và mang lại khả năng giải thích cho mạng nơ-ron, dựa trên Attention ta có thể hiểu được cơ chế làm việc của mạng nơ-ron trong học sâu.

Mặc dù đã cố gắng hết sức nhưng do trình độ chuyên môn, thời gian có hạn và không có cơ sở vật chất tốt về phần cứng nên đồ án còn những hạn chế nhất định như: chưa tinh chỉnh được nhiều siêu tham số, chưa thử nghiệm được nhiều cơ chế Attention khác nhau, chưa huấn luyện được trên tập dữ liệu lớn,... Do đó chưa thực sự đem lại được kết quả như mong muốn.

### 4.2 Định hướng phát triển

- Huấn luyện mô hình trên tập dữ liệu lớn hơn như là MS CoCo.
- Tiếp tục nghiên cứu và tối ưu bài toán với cơ chế học tăng cường và dùng độ đo Rough để đánh giá hàm lỗi.
- Triển khai thành một ứng dụng mô tả ảnh trong thực tế bằng tiếng việt trên hệ điều hành android.

## DANH MỤC TÀI LIỆU THAM KHẢO

- [1] Kelvin Xu, Jimmy Lei Ba, Ryan Kiros, Kyunghyn Cho, Aaron Courville, Ruslan Salakhutdinov, Richard S.Zemel, Yoshua Bengio, Show Attend and Tell: Neural Image Caption Generation with Visual Attention, 19 Apr 2016.
- [2] Oriol Cinyals, Alexander Toshev, Samy Bengio, Dumitru Erhan, Show and Tell: A Neural Image Caption Generator, 20 Apr 2015.
- [3] Andrej Karpathy, Fei-Fei Li, Automated Image Captioning with ConvNets and Recurrent Nets.
- [4] Bahdanau, Dzmitry, Bengio, Yoshua, Neural Machine Translation By Jointly Learning To Align And Translate, 2014.
- [5] Afshine Amidi, Shervine Amidi, CS229 - Machine Learning.
- [6] Afshine Amidi, Shervine Amidi, CS230 - Deep Learning.
- [7] Stanford, CS231n, Convolutional Neural Networks for Visual Recognition.
- [8] N. Arbel, How LSTM networks solve the problem of vanishing gradients, 2018.
- [9] Heuritech, Attention mechanism, 2019.
- [10] V. H. Tiệp, Machine Learning Cơ Bản.
- [11] N. T. Tuấn, Deep Learning Cơ Bản.