TRƯỜNG ĐẠI HỌC CÔNG NGHỆ THÔNG TIN &

TRUYỀN THÔNG VIỆT HÀN

Khoa Khoa Học Máy Tính



BÁO CÁO ĐỒ ÁN MÔN HỌC SÂU

**XÂY DỰNG MÔ HÌNH SINH MÔ TẢ CHO HÌNH ẢNH**

|  |  |  |
| --- | --- | --- |
| Sinh viên thực hiện: | Lê Văn Phước | 20IT487 |
|  | Hồ Luận | 20IT669 |
|  | Võ Văn Nguyên | 20IT325 |
|  | Nguyễn Khắc Đông | 20IT146 |
|  | Lê Đức Hiệu | 20IT369 |
| Giảng viên hướng dẫn: | ThS. Trần Uyên Trang | |

Đà Nẵng, 05 tháng 01 năm 2023

TRƯỜNG ĐẠI HỌC CÔNG NGHỆ THÔNG TIN &

THÔNG VIỆT HÀN

Khoa Khoa Học Máy Tính



BÁO CÁO ĐỒ ÁN MÔN HỌC SÂU

**XÂY DỰNG MÔ HÌNH SINH MÔ TẢ CHO HÌNH ẢNH**

|  |  |  |
| --- | --- | --- |
| Sinh viên thực hiện: | Lê Văn Phước | 20IT487 |
|  | Hồ Luận | 20IT669 |
|  | Võ Văn Nguyên | 20IT325 |
|  | Nguyễn Khắc Đông | 20IT146 |
|  | Lê Đức Hiệu | 20IT369 |
| Giảng viên hướng dẫn: | ThS. Trần Uyên Trang | |

Đà Nẵng, 05 tháng 01 năm 2023

**NHẬN XÉT CỦA GIẢNG VIÊN HƯỚNG DẪN**

|  |  |
| --- | --- |
|  | Chữ ký của giảng viên  *(Ký và ghi rõ họ tên)* |

**LỜI CẢM ƠN**

Đồ án môn học “Học sâu” lần này được hoàn thành tại trường Đại học Công nghệ thông tin và Truyền thông Việt-Hàn. Trong quá trình học tập, triển khai và hoàn thành đồ án, chúng em đã nhận được rất nhiều sự giúp đỡ, vì vậy chúng em muốn viết lời cảm ơn này đến:

Chúng em xin gửi lời cảm ơn chân thành tới thầy cô trường Đại học Công nghệ thông tin và Truyền thông Việt Hàn đã giảng dạy và nâng đỡ cho chúng em thời gian qua. Đặc biệt là ThS. Trần Uyên Trang là người trực tiếp chỉ bảo và hướng dẫn chúng em hoàn thành đồ án này.

Bên cạnh đó để hoàn thành đồ án này chúng em đã nhận được những lời khuyên sự giúp đỡ tận tình từ anh chị em bạn bè, chúng em xin cảm ơn.

Mặc dù đã vận dụng tất cả kiến thức đã được học tập và kinh nghiệm thực tế từ bản thân để hoàn thành đồ án này, song có thể còn có những mặt hạn chế, thiếu sót. Chúng em rất mong nhận được ý kiến đóng góp và sự chỉ dẫn của các thầy cô giáo. Chúng em xin chân thành cảm ơn.

**MỤC LỤC**

[DANH MỤC HÌNH ẢNH 6](#_Toc155933114)

[DANH MỤC CÁC TỪ VIẾT TẮT 7](#_Toc155933115)

[MỞ ĐẦU 8](#_Toc155933116)

[1. Lý do chọn đề tài 8](#_Toc155933117)

[2. Mục tiêu của đề tài 8](#_Toc155933118)

[3. Nội dung và kế hoạch thực hiện 8](#_Toc155933119)

[5. Cấu trúc đề tài 9](#_Toc155933120)

[Chương 1. TỔNG QUAN VỀ ĐỀ TÀI 10](#_Toc155933121)

[1. Giới thiệu chung về đề tài 10](#_Toc155933122)

[2. Tình hình nghiên cứu trong nước và ngoài nước 10](#_Toc155933123)

[3 Đối tượng và phạm nghiên cứu 10](#_Toc155933124)

[4. Phương pháp nghiên cứu 10](#_Toc155933125)

[Chương 2: CƠ SỞ LÝ THUYẾT 12](#_Toc155933126)

[1. Ngôn ngữ lập trình Python 12](#_Toc155933127)

[2. Mạng CNN 12](#_Toc155933128)

[3. LSTM 15](#_Toc155933129)

[4. Word embedding Glove 16](#_Toc155933130)

[5. Hàm đánh giá BLEU 16](#_Toc155933131)

[Chương 3: XÂY DỰNG MÔ HÌNH 18](#_Toc155933132)

[3.1 Tập dữ liệu Flickr8k 18](#_Toc155933133)

[3.2 Tiền xử lý dữ liệu 18](#_Toc155933134)

[3.3 Trích xuất đặc trưng hình ảnh 19](#_Toc155933135)

[3.4. Tạo dữ liệu huấn luyện 20](#_Toc155933136)

[3.5 Xây dựng mô hình 1 22](#_Toc155933137)

[3.6 Xây dựng mô hình 2 24](#_Toc155933138)

[3.7 Đánh giá độ chính xác 25](#_Toc155933139)

[KẾT LUẬN 27](#_Toc155933140)

[DANH MỤC TÀI LIỆU THAM KHẢO 28](#_Toc155933141)

# DANH MỤC HÌNH ẢNH

[Hình 1. Ngôn ngữ lập trình Python 12](#_Toc155933142)

[Hình 2. Kiến trúc cơ bản của mạng CNN 13](#_Toc155933143)

[Hình 3. Bộ lọc tích chập 13](#_Toc155933144)

[Hình 4. Phương pháp Average Pooling và Max Pooling 14](#_Toc155933145)

[Hình 5. Cấu trúc LSTM 15](#_Toc155933146)

[Hình 6. Word embedding 16](#_Toc155933147)

[Hình 7. Hàm BLEU được dử dụng rỗng rãi trong dịch máy 17](#_Toc155933148)

[Hình 8. Tập dữ liệu Flickr8k 18](#_Toc155933149)

[Hình 9. Mô tả dữ liệu traning 20](#_Toc155933150)

[Hình 10. Cấu túc của mô hình 1 23](#_Toc155933151)

[Hình 11. Cấu trúc mô hình 2 24](#_Toc155933152)

[Hình 12. Độ chính xác của mô hình theo hàm BLEU 26](#_Toc155933153)

# DANH MỤC CÁC TỪ VIẾT TẮT

|  |  |
| --- | --- |
| Cụm từ viết tắt | Cụm từ đầy đủ |
| LSTM | Long short-term memory |
| BLUE | Cơ sở dữ liệu |
|  | Người sử dụng |
| CNTT&TT | Công nghệ thông tin và truyền thông |

# MỞ ĐẦU

## 1. Lý do chọn đề tài

Trong thời đại số hóa ngày nay, hình ảnh đóng vai trò trung tâm trong việc truyền tải thông tin và kiến thức. Sự phát triển của công nghệ máy ảnh và thiết bị di động đã làm tăng lượng hình ảnh được tạo ra và chia sẻ một cách chưa từng có. Điều này tạo ra một nhu cầu lớn cho việc tự động hóa quá trình mô tả và hiểu hình ảnh, không chỉ trong lĩnh vực trí tuệ nhân tạo mà còn trong các ứng dụng thực tế như hỗ trợ người khiếm thị, quản lý và tìm kiếm hình ảnh, và tạo ra nội dung trực quan hấp dẫn.

Chính vì lý do này, đề tài “Xây dựng mô hình sinh mô tả hình ảnh” được chọn để nghiên cứu nhằm đáp ứng nhu cầu đó và đóng góp vào sự tiến bộ của khoa học máy tính. Mục tiêu là phát triển một mô hình có khả năng tạo ra mô tả chính xác và tự nhiên cho hình ảnh, giúp cải thiện khả năng tương tác giữa con người và máy tính, cũng như mở rộng khả năng ứng dụng của trí tuệ nhân tạo trong đời sống hàng ngày.

## 2. Mục tiêu của đề tài

Mục tiêu chính của đề tài này là phát triển một mô hình trí tuệ nhân tạo có khả năng tự động sinh ra mô tả ngôn ngữ tự nhiên cho hình ảnh. Điều này bao gồm việc:

* Hiểu và phân tích nội dung hình ảnh
* Tối ưu hóa hiệu suất mô hình
* Ứng dụng trong thực tế

Ngoài ra, mục tiêu phụ của đề tài cũng bao gồm việc khám phá các kỹ thuật mới trong lĩnh vực học máy và thị giác máy tính, cũng như đóng góp vào cộng đồng khoa học thông qua việc chia sẻ kiến thức và kết quả nghiên cứu. Đề tài này không chỉ nhằm mục đích giải quyết các vấn đề cụ thể mà còn hướng đến việc mở rộng hiểu biết về cách thức mà máy móc có thể “nhìn” và “hiểu” thế giới xung quanh chúng ta.

## 3. Nội dung và kế hoạch thực hiện

Các nội dung sẽ thực hiện trong đề tài:

- Thu nhập và xử lý dữ liệu.

- Xây dựng mô hình sinh mô tả cho hình ảnh.

- Đánh giá hiệu suất của mô hình và tối ưu hóa.

- Xây dựng ứng dụng sinh mô tả cho hình ảnh.

Kế hoạch thực hiện:

|  |  |
| --- | --- |
| Thời gian | Nội dung thực hiện |
| Tuần thứ 1 (từ 21/10 đến 27/10 ) | Tìm hiểu đề tài |
| Tuần thứ 2, 3 (từ 27/10 đến 11/11) | Thu nhập và xử lý dữ liệu |
| Tuần thứ 4, 5 (từ 11/11 đến 26/11) | Xây dựng mô hình học sâu |
| Tuần thứ 6 (từ 26/11 đến 03/12) | Đánh giá hiệu suất của mô hình và tối ưu hóa. |
| Tuần thứ 7 (từ 03/12 đến 10/12) | Triển khai mô hình lên website |
| Tuần thứ 8 (từ 10/12 đến 14/12) | Hoàn thiện slide và báo cáo. |

## 5. Cấu trúc đề tài

Sau phần *Mở đầu*, báo cáo được trình bày trong 3 chương, cụ thể như sau:

- Chương 1: *Tổng quan về đề tài*. Trong chương này, báo cáo trình bày các khái niệm, các kiến thức cơ bản về đề tài.

- Chương 2: *Cơ sở lý thuyết*. Nội dung chương bao gồm các lý thuyết thực hiện trong đề tài.

- Chương 3: *Xây dựng mô hình*. Chương đề xuất xây dựng một mô hình học sâu để có thể giải quyết được bài toán.

Cuối cùng là *Kết luận*, *Tài liệu tham khảo* và *Phụ lục* liên quan đến đề tài.

# Chương 1. TỔNG QUAN VỀ ĐỀ TÀI

## 1. Giới thiệu chung về đề tài

Trong những năm gần đây, sự phát triển nhanh chóng của trí tuệ nhân tạo đã khiến cho việc tạo mô tả hình ảnh tự động trở nên thu hút sự chú ý của nhiều nhà nghiên cứu trong lĩnh vực trí tuệ nhân tạo và trở thành một nhiệm vụ thú vị và khó khăn. [Mô hình sinh mô tả hình ảnh tự động tạo ra các mô tả ngôn ngữ tự nhiên dựa trên nội dung quan sát được trong hình ảnh, là một phần quan trọng của việc hiểu cảnh, kết hợp kiến thức về thị giác máy tính và xử lý ngôn ngữ tự nhiê](https://www.hindawi.com/journals/cin/2020/3062706/)n.

[Các phương pháp tạo mô tả hình ảnh đã phát triển từ các cách tiếp cận dựa trên học máy truyền thống sang các phương pháp dựa trên học sâu, đã trở nên thống trị do khả năng hiệu quả trong việc trích xuất đặc trưng hình ản](https://www.hindawi.com/journals/cin/2020/3062706/)h. [Các nghiên cứu cũng đã khám phá hai cách tiếp cận chính cho việc tạo mô tả hình ảnh: tạo mô tả dày đặc (dựa trên khu vực) và tạo mô tả toàn cản](https://www.hindawi.com/journals/cin/2020/3062706/)h.

## 2. Tình hình nghiên cứu trong nước và ngoài nước

Tình hình nghiên cứu trong nước: Nghiên cứu về mô hình sinh mô tả hình ảnh trong nước đã có những bước tiến quan trọng. Các nhà nghiên cứu Việt Nam đã tập trung vào việc phát triển các mô hình ngôn ngữ và thị giác máy tính phù hợp với ngữ cảnh và dữ liệu đặc thù của Việt Nam..

Tình hình nghiên cứu ngoài nước: Ở cấp độ quốc tế, nghiên cứu về mô hình sinh mô tả hình ảnh đã đạt được nhiều thành tựu đáng kể. Các nhà nghiên cứu quốc tế đã phát triển các mô hình học sâu tiên tiến, nhằm cải thiện khả năng hiểu và mô tả hình ảnh của máy móc. Các nghiên cứu này không chỉ giúp cải thiện hiệu suất của các mô hình mà còn mở ra hướng đi mới cho việc tích hợp thị giác máy tính và xử lý ngôn ngữ tự nhiên.

## 3 Đối tượng và phạm nghiên cứu

Về đối tượng nghiên cứu:

* Hình ảnh: Các hình ảnh đa dạng từ nhiều nguồn khác nhau, bao gồm hình ảnh tự nhiên, hình ảnh từ các cơ sở dữ liệu công cộng, và hình ảnh từ mạng xã hội.
* Mô hình trí tuệ nhân tạo: Các mô hình học sâu và thuật toán xử lý ngôn ngữ tự nhiên hiện đại, với trọng tâm là cơ chế chú ý và mạng nơ-ron tích chập.

Về phạm vi nghiên cứu:

* Phát triển mô hình: Tập trung vào việc thiết kế và huấn luyện mô hình có khả năng sinh mô tả ngôn ngữ tự nhiên cho hình ảnh.
* Đánh giá hiệu suất: Sử dụng các bộ dữ liệu chuẩn và tiêu chí đánh giá đã được công nhận để đánh giá khả năng của mô hình.
* Ứng dụng thực tế: Khám phá khả năng áp dụng mô hình trong các tình huống thực tế và các lĩnh vực khác nhau

## 4. Phương pháp nghiên cứu

Đề tài này tập trung vào việc xây dựng mô hình giúp sinh mô tả cho hình ảnh. Phương pháp nghiên cứu chính sẽ là phân tích dữ liệu và học sâu.

Dữ liệu ảnh và văn bản mô tả sẽ được thu nhập từ các nguồn khác nhau. Các thuật toán học sâu sẽ được sử dụng để phân tích và xử lý dữ liệu. Việc lựa chọn mô hình sẽ dựa trên độ chính xác của mô hình trong tập dữ liệu kiểm tra.

*Kết chương 1: Chúng ta đã có cái nhìn tổng quan về đề tài sinh mô tả cho hình ảnh và sự cần thiết của đề tài. Tiếp theo, ta sẽ tiếp tục với cơ sở lý thuyết được trình bày trong chương 2.*

# Chương 2: CƠ SỞ LÝ THUYẾT

## 1. Ngôn ngữ lập trình Python

Python là một ngôn ngữ lập trình phổ biến, đa mục đích và dễ học. Python có cú pháp đơn giản, rõ ràng và linh hoạt, cho phép lập trình viên viết các chương trình ngắn gọn và hiệu quả. Python hỗ trợ nhiều lập trình hướng đối tượng, thủ tục, hàm và lập trình hướng khía cạnh. Python cũng có một thư viện chuẩn phong phú, cung cấp các công cụ cho nhiều lĩnh vực như xử lý chuỗi, toán học, đồ họa, mạng, web, khoa học dữ liệu và trí tuệ nhân tạo. Python có thể chạy trên nhiều hệ điều hành khác nhau, bao gồm Windows, Linux, Mac OS và Android . Python là một ngôn ngữ lập trình động, có nghĩa là các biến không cần được khai báo trước khi sử dụng và các kiểu dữ liệu có thể thay đổi theo thời gian. Python cũng hỗ trợ nhiều cơ chế quản lý bộ nhớ tự động, giúp giảm thiểu lỗi và tăng hiệu suất.

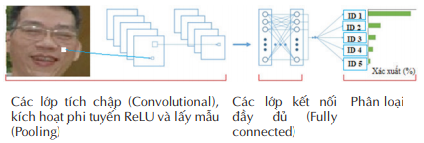


Hình 1. Ngôn ngữ lập trình Python

Python được tạo ra bởi Guido van Rossum vào năm 1991, với mục tiêu làm cho ngôn ngữ lập trình dễ đọc hơn và thú vị hơn. Python có một cộng đồng lập trình viên lớn và năng động, đóng góp nhiều mã nguồn mở, tài liệu và hướng dẫn cho ngôn ngữ. Python cũng có nhiều khung và thư viện bên ngoài, như Django, Flask, NumPy, Pandas, TensorFlow và PyTorch, mở rộng khả năng của ngôn ngữ và làm cho nó phù hợp với nhiều ứng dụng khác nhau.

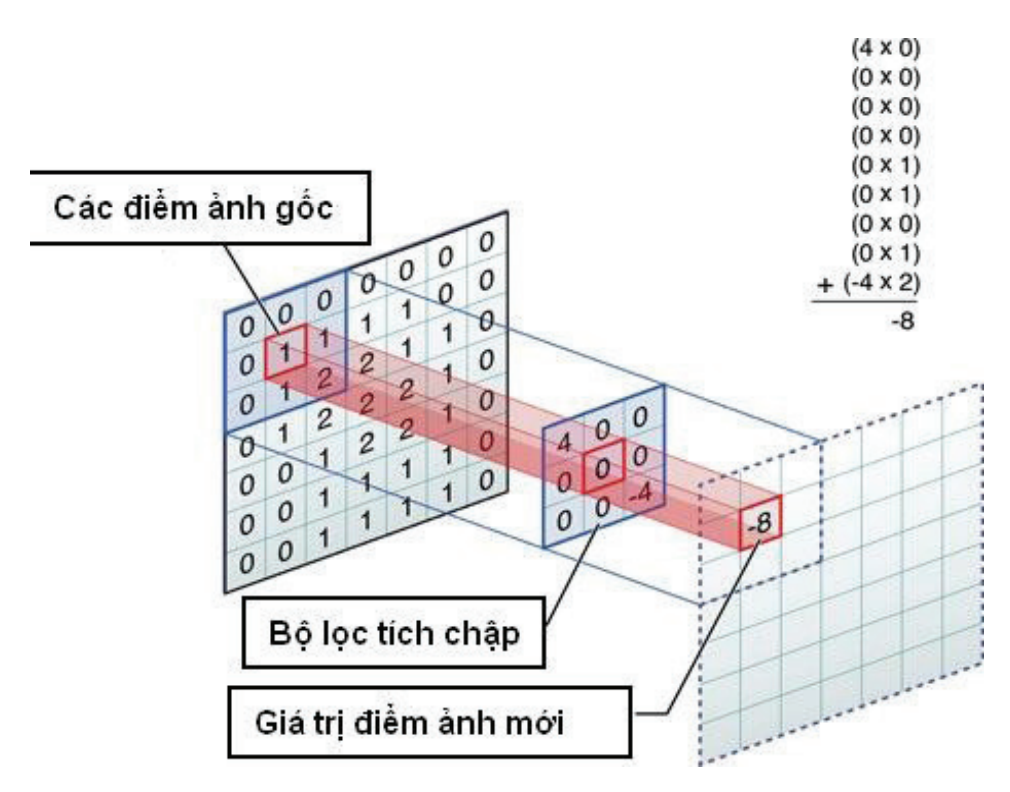
## 2. Mạng CNN

Hình dưới trình bày một kiến trúc mạng CNN, các lớp cơ bản trong một mạng CNN bao gồm: lớp tích chập (Convolutional); lớp kích hoạt phi tuyến ReLU (Rectified Linear Unit); lớp lấy mẫu (Pooling); lớp kết nối đầy đủ (Fully connected) được thay đổi về số lượng và cách sắp xếp để tạo ra các mô hình huấn luyện phù hợp cho từng bài toán khác.



Hình 2. Kiến trúc cơ bản của mạng CNN

- Lớp tích chập: đây là thành phần quan trọng nhất trong mạng CNN, thể hiện sự liên kết cục bộ thay vì kết nối toàn bộ các điểm ảnh. Các liên kết cục bộ được tính toán bằng phép tích chập giữa các giá trị điểm ảnh trong một vùng ảnh cục bộ với các bộ lọc filters có kích thước nhỏ.



Hình 3. Bộ lọc tích chập

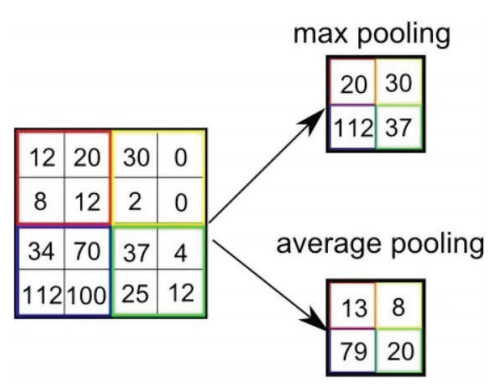
Bộ lọc tích chập được sử dụng trên ma trận điểm ảnh. Trong hình 2.2, bộ lọc được sử dụng là một ma trận có kích thước 3x3, bộ lọc này dịch chuyển lần lượt qua từng vùng ảnh đến khi hoàn thành quét toàn bộ bức ảnh, tạo ra một bức ảnh mới có kích thước nhỏ hơn hoặc bằng với kích thước ảnh đầu vào. Kích thước này được quyết định tùy theo kích thước các khoảng trắng được thêm ở viền bức ảnh gốc và được tính theo công thức sau:

Trong đó: O: kích thước ảnh đầu ra; i: kích thước ảnh đầu vào; p: kích thước khoảng trắng phía ngoài viền của ảnh gốc; k: kích thước bộ lọc; s: bước trượt của bộ lọc.

Như vậy, sau khi đưa một bức ảnh đầu vào cho lớp tích chập nhận được kết quả đầu ra là một loạt ảnh tương ứng với các bộ lọc đã được sử dụng để thực hiện phép tích chập. Các trọng số của các bộ lọc này được khởi tạo ngẫu nhiên trong lần đầu tiên và sẽ được cập nhật trong quá trình huấn luyện.

- Lớp kích hoạt phi tuyến ReLU: được xây dựng để đảm bảo tính phi tuyến của mô hình huấn luyện sau khi đã thực hiện một loạt các phép tính toán tuyến tính qua các lớp tích chập. Lớp kích hoạt phi tuyến sử dụng các hàm kích hoạt phi tuyến như ReLU hoặc sigmoid, tanh… để giới hạn phạm vi biên độ cho phép của giá trị đầu ra. Trong số các hàm kích hoạt này, hàm ReLU được chọn do cài đặt đơn giản, tốc độ xử lý nhanh mà vẫn đảm bảo được tính toán hiệu quả. Phép tính toán của hàm ReLU chỉ đơn giản là chuyển tất cả các giá trị âm thành giá trị 0. Lớp ReLU được áp dụng ngay phía sau lớp tích chập, với đầu ra là một ảnh mới có kích thước giống với ảnh đầu vào, các giá trị điểm ảnh cũng hoàn toàn tương tự, trừ các giá trị âm đã bị loại bỏ.

- Lớp lấy mẫu: được đặt sau lớp tích chập và lớp ReLU để làm giảm kích thước ảnh đầu ra trong khi vẫn giữ được các thông tin quan trọng của ảnh đầu vào. Việc giảm kích thước dữ liệu có tác dụng làm giảm được số lượng tham số cũng như tăng hiệu quả tính toán. Lớp lấy mẫu cũng sử dụng một cửa sổ trượt để quét toàn bộ các vùng trong ảnh như lớp tích chập, và thực hiện phép lấy mẫu thay vì phép tích chập, sẽ chọn lưu lại một giá trị duy nhất đại diện cho toàn bộ thông tin của vùng ảnh đó.



Hình 4. Phương pháp Average Pooling và Max Pooling

Như vậy, với mỗi ảnh đầu vào được đưa qua lấy mẫu sẽ thu được một ảnh đầu ra tương ứng, có kích thước giảm xuống đáng kể nhưng vẫn giữ được các đặc trưng cần thiết cho quá trình tính toán và nhận dạng.

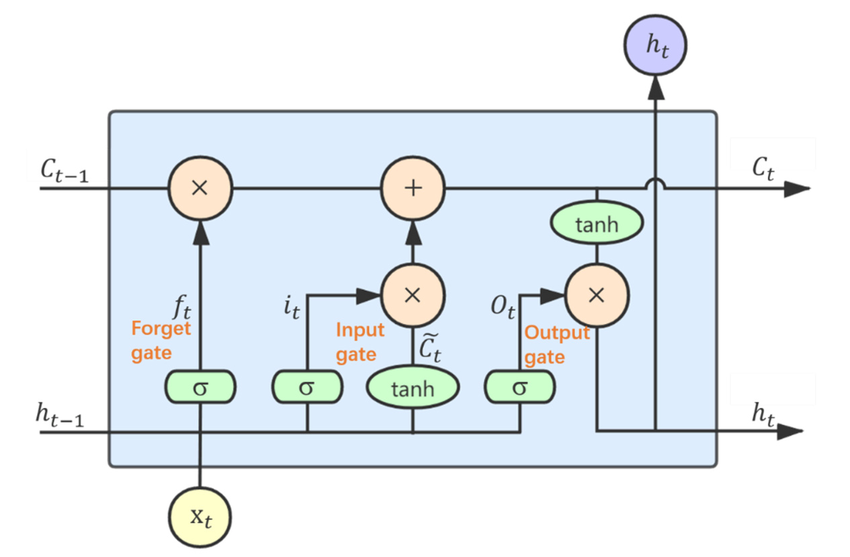
- Lớp kết nối đầy đủ: được thiết kế tương tự như trong mạng nơ ron truyền thống, tất cả các điểm ảnh được kết nối đầy đủ với node trong lớp tiếp theo.

So với mạng nơ ron truyền thống, các ảnh đầu vào của lớp này đã có kích thước được giảm bớt rất nhiều, đồng thời vẫn đảm bảo các thông tin quan trọng của ảnh cho việc nhận dạng. Do vậy, việc tính toán nhận dạng sử dụng mô hình truyền thẳng đã không còn phức tạp và tốn nhiều thời gian như trong mạng nơ ron truyền thống.

## 3. LSTM

LSTM là một kiến trúc mạng thần kinh hồi quy (RNN) nhân tạo, được sử dụng để xử lý các dữ liệu tuần tự, như lời nói, văn bản, video, hoặc chuỗi thời gian. LSTM có thể học được các mối quan hệ giữa các phần tử trong chuỗi, không chỉ dựa trên các phần tử trước đó, mà còn dựa trên các phần tử sau đó. Điều này giúp cho LSTM có thể nắm bắt được ngữ cảnh của chuỗi một cách tốt hơn.

Lớp LSTM có một tế bào (cell), một cổng vào (input gate), một cổng ra (output gate), và một cổng quên (forget gate). Tế bào ghi nhớ các giá trị trong các khoảng thời gian bất kỳ, và ba cổng điều chỉnh luồng thông tin ra/vào tế bào. Các cổng sử dụng các hàm kích hoạt như sigmoid hoặc tanh để quyết định xem nên giữ hay bỏ đi những thông tin nào.



Hình 5. Cấu trúc LSTM

Cách thức hoạt động của LSTM có thể được minh họa như sau:

- Cho một chuỗi đầu vào X = (x1, x2, …, xn), mỗi phần tử xi là một vector đặc trưng của một đơn vị tuần tự, như một từ, một âm thanh, hoặc một khung hình.

- Lớp LSTM từ trái sang phải sẽ tính toán các trạng thái ẩn (hidden states) hi (i = 1, 2, …, n) theo công thức:

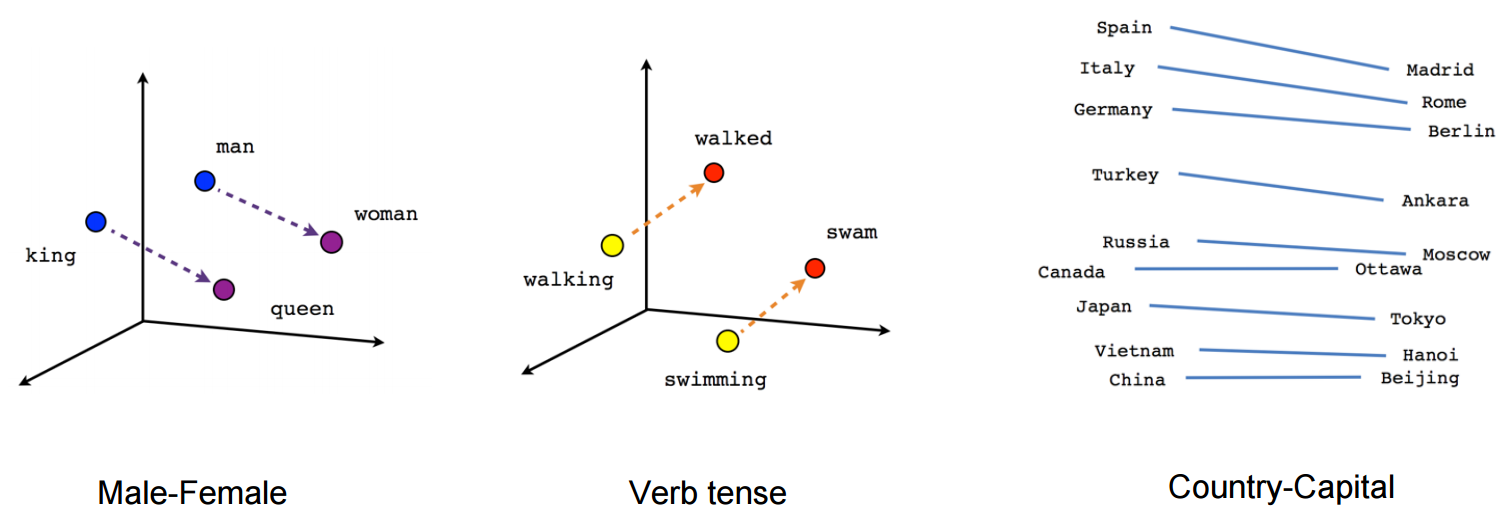
* , đây là cổng quên, quyết định xem nên giữ lại bao nhiêu thông tin từ trạng thái ẩn trước đó và tế bào trước đó.
* , đây là cổng vào, quyết định xem nên thêm vào bao nhiêu thông tin mới từ đầu vào hiện tại và trạng thái ẩn trước đó.
* , đây là tế bào, cập nhật giá trị của tế bào dựa trên cổng quên và cổng vào.
* , cổng ra, quyết định xem nên đưa ra bao nhiêu thông tin từ tế bào hiện tại và trạng thái ẩn trước đó.
* , trạng thái ẩn, đại diện cho thông tin đã học được từ các đầu vào trước đó từ trái sang phải.

## 4. Word embedding Glove

Word Embedding là một kỹ thuật quan trọng trong xử lý ngôn ngữ tự nhiên (NLP) giúp biến đổi từ vựng thành các vector số học có thể được xử lý bởi các mô hình máy học. [Glove, viết tắt của “Global Vectors for Word Representation”, là một thuật toán học không giám sát được phát triển bởi nhóm nghiên cứu tại Đại học Stanford](https://nlp.stanford.edu/projects/glove/). [Glove nhằm mục đích thu được các biểu diễn vector cho từ vựng dựa trên thống kê đồng xuất hiện toàn cầu từ một bộ sưu tập văn bản, và các biểu diễn này thể hiện các cấu trúc tuyến tính thú vị trong không gian vector từ vựng](https://nlp.stanford.edu/projects/glove/).

**Cách hoạt động của Glove**: Glove xây dựng một ma trận đồng xuất hiện, nắm bắt thông tin về tần suất xuất hiện cùng nhau của các cặp từ trong một khoảng cách cố định. Sau đó, thuật toán tối ưu hóa các vector từ để chúng phản ánh thông tin đồng xuất hiện này. [Mục tiêu là làm cho tích vô hướng của hai vector từ tương ứng với logarit của xác suất đồng xuất hiện của chúng, qua đó bắt được các mối quan hệ ngữ nghĩa giữa các từ](https://www.geeksforgeeks.org/pre-trained-word-embedding-using-glove-in-nlp-models/).

Glove được sử dụng rộng rãi trong nhiều tác vụ NLP khác nhau, từ phân loại văn bản đến nhận dạng thực thể có tên. [Các vector từ Glove cung cấp một cách hiệu quả để biểu diễn ngữ nghĩa của từ vựng, giúp cải thiện hiệu suất của các mô hình học máy và học sâu trong việc xử lý dữ liệu văn bản](https://paperswithcode.com/method/glove).



Hình 6. Word embedding

## 5. Hàm đánh giá BLEU

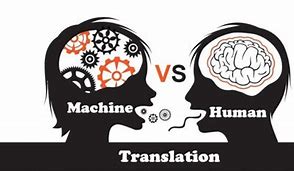
Trong bài toán sinh mô tả hình ảnh, một hình ảnh có thể được gán nhiều nhãn khác nhau. Vì vậy, các phương pháp đánh giá phổ biến như accuracy, F1,... không thể sử dụng được. Ví dụ:

- Nhãn mục tiêu: “Cô gái đang ngồi trên ghế đá”.

- Nhãn dự đoán: “Cô gái mặc áo đỏ đang ngồi trên ghế đá tại công viên”

Rõ ràng, hai nhãn là khác nhau nhưng chất lượng nội dung mô tả vẫn đúng. Vì vậy, ta cần một hàm đánh giá được mức độ giống nhau giữa nhãn mục tiêu và nhãn dự đoán. Thật may mắn, hàm BLEU có thể làm được điều đó.

Hàm đánh giá BLEU (Bilingual Evaluation Understudy) là một phương pháp đánh giá tự động được sử dụng rộng rãi để đo lường chất lượng của văn bản được dịch bởi máy so với một hoặc nhiều bản dịch tham chiếu do con người tạo ra. BLEU được thiết kế để đánh giá độ chính xác của dịch máy ở cấp độ cụm từ thông qua việc so sánh sự trùng khớp của các n-gram giữa bản dịch và bản dịch tham chiếu.



Hình 7. Hàm BLEU được dử dụng rỗng rãi trong dịch máy

Cách tính điểm BLEU: Điểm BLEU được tính dựa trên hai yếu tố chính:

* Precision: Tính toán tỷ lệ các n-gram trong bản dịch máy xuất hiện trong bản dịch tham chiếu.
* Brevity Penalty (BP): Phạt độ ngắn của bản dịch máy so với bản dịch tham chiếu để tránh ưu tiên cho các bản dịch quá ngắn.

Công thức tính điểm BLEU là:

trong đó:

* pn là precision của n-gram.
* wn là trọng số của mỗi n-gram (thường được chọn để mỗi trọng số bằng nhau).
* N là số lượng n-gram tối đa được xem xét.
* BP là brevity penalty, được tính bằng công thức:

trong đó:

c là độ dài của bản dịch máy.

r là độ dài của bản dịch tham chiếu.

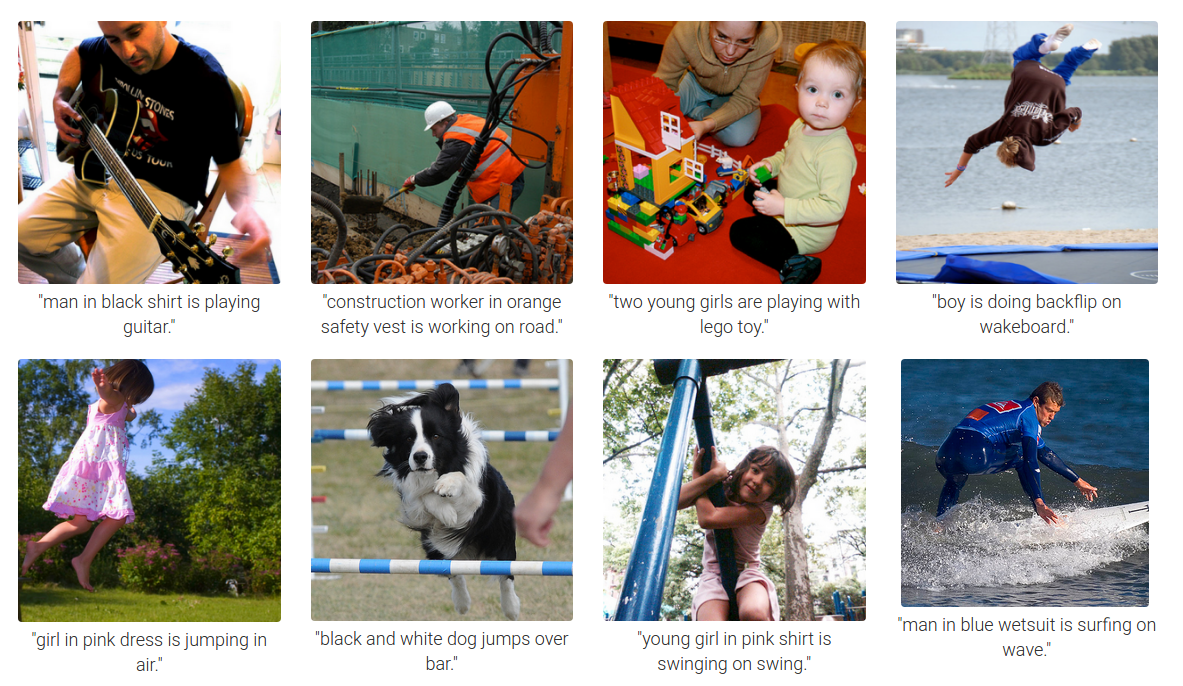
# Chương 3: XÂY DỰNG MÔ HÌNH

## 3.1 Tập dữ liệu Flickr8k

Tập dữ liệu Flickr8k là một bộ sưu tập hình ảnh được sử dụng rộng rãi trong nghiên cứu về xử lý ngôn ngữ tự nhiên và thị giác máy tính, đặc biệt là trong lĩnh vực sinh mô tả hình ảnh (image captioning). [Bộ dữ liệu này bao gồm 8.000 hình ảnh thu thập từ trang web chia sẻ hình ảnh Flickr, mỗi hình ảnh đi kèm với 5 mô tả bằng văn bản do con người tạo ra](https://www.kaggle.com/datasets/adityajn105/flickr8k).

**Đặc điểm của tập dữ liệu Flickr8k:**

* **Đa dạng**: Hình ảnh trong tập dữ liệu đại diện cho một loạt các tình huống, đối tượng và hoạt động, cung cấp một nguồn dữ liệu phong phú cho việc huấn luyện và kiểm tra các mô hình.
* **Mô tả chất lượng cao**: Mỗi hình ảnh được mô tả bởi 5 câu khác nhau, giúp cung cấp một cái nhìn đa chiều về nội dung hình ảnh.
* **Ứng dụng thực tế**: Tập dữ liệu được sử dụng để phát triển các mô hình có khả năng sinh mô tả tự động cho hình ảnh, một bước quan trọng trong việc cải thiện tương tác giữa con người và máy tính.



Hình 8. Tập dữ liệu Flickr8k

## 3.2 Tiền xử lý dữ liệu

Tiền xử lý dữ liệu là bước quan trọng trong quá trình xây dựng mô hình. Dữ liệu phải được xử lý sạch sẽ, loại bỏ các chi tiết thừa và định dạng lại cho phù hợp với yêu cầu bài toán. Ở đây, chúng tôi thực hiện:

* Chuyển các câu thành chữ thường
* Loại bỏ ký tự đặc biệt và số có mặt trong văn bản
* Loại bỏ các khoảng trắng dư thừa
* Loại bỏ các ký tự đơn
* Thêm thẻ bắt đầu và thẻ kết thúc vào các câu để chỉ ra điểm bắt đầu và kết thúc của một câu

def text\_preprocessing(data):

    # Chuyển các ký tự trong cột 'caption' về chữ thường

    data['caption'] = data['caption'].apply(lambda x: x.lower())

    # Loại bỏ các ký tự không phải là chữ cái trong cột 'caption'

    data['caption'] = data['caption'].apply(lambda x: x.replace("[^A-Za-z]",""))

    # Loại bỏ các khoảng trắng dư thừa trong cột 'caption'

    data['caption'] = data['caption'].apply(lambda x: x.replace("\s+"," "))

    # Loại bỏ các từ có độ dài là 1 trong cột 'caption'

    data['caption'] = data['caption'].apply(lambda x: " ".join([word for word in x.split() if len(word)>1]))

    # Thêm "startseq" vào đầu và "endseq" vào cuối mỗi câu trong cột 'caption'

    data['caption'] = "startseq "+data['caption']+" endseq"

    return data

Tiếp theo, ta sẽ thực hiện tokenize mã hóa văn bản thành các số để tiện cho việc sử dụng sau này.

 Tạo một tokenizer để mã hóa văn bản thành các chuỗi số

tokenizer = Tokenizer()

tokenizer.fit\_on\_texts(captions)

vocab\_size = len(tokenizer.word\_index) + 1

max\_length = max(len(caption.split()) for caption in captions)

images = captions\_df['image'].unique()

nimages = len(images)

# # Phân chia tập hình ảnh thành tập huấn luyện và tập validation

split\_index = round(0.85\*nimages)

train\_images = images[:split\_index]

val\_images = images[split\_index:]

train = captions\_df[captions\_df['image'].isin(train\_images)]

test = captions\_df[captions\_df['image'].isin(val\_images)]

train.reset\_index(inplace=True,drop=True)

test.reset\_index(inplace=True,drop=True)

## 3.3 Trích xuất đặc trưng hình ảnh

Mô hình DenseNet201 là một kiến trúc mạng nơ-ron tích chập sâu (CNN) được biết đến với cấu trúc “kết nối dày đặc” của nó. Cấu trúc kết nối dày đặc là nơi mỗi lớp nhận đầu vào từ tất cả các lớp trước đó và truyền đầu ra của nó đến tất cả các lớp sau. [Điều này tạo ra một mô hình có khả năng học các đặc trưng phức tạp mà không cần quá nhiều tham số, giúp giảm nguy cơ overfitting và tăng hiệu suất tính toán](https://www.phamduytung.com/blog/2018-12-11-a-bunch-of-tips-and-tricks-for-training-deep-neural-networks/). Ta sử dụng mô hình pretrained này để trích xuất đặc trưng của hình ảnh và lưu kết quả vào biến features.

## 3.4. Tạo dữ liệu huấn luyện

# Khởi tạo một mô hình DenseNet201

model = DenseNet201()

fe = Model(inputs=model.input, outputs=model.layers[-2].output)

img\_size = 224

features = {}

for image in tqdm(captions\_df['image'].unique().tolist()):

    img = load\_img(os.path.join(image\_path,image),target\_size=(img\_size,img\_size))

    img = img\_to\_array(img)

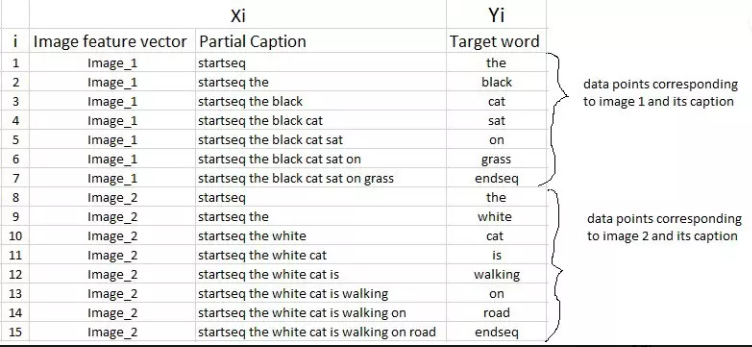
    img = img/255.

    img = np.expand\_dims(img,axis=0)

    feature = fe.predict(img, verbose=0)

    features[image] = feature

Dữ liệu gồm 2 phần là chuỗi mô tả và hình ảnh. Chuỗi mô tả bắt đầu bằng startseq và kết thúc bằng endseq. Một điểm dữ liệu trong tập train sẽ có dạng ảnh + từ và kết quả dự đoán là từ tiếp theo trong câu. Quá trình dự đoán sẽ kết thúc nếu từ dự đoán là “endseq”.



Hình 9. Mô tả dữ liệu traning

class CustomDataGenerator(Sequence):

    # Khởi tạo hàm constructor với các tham số được truyền vào.

    def \_\_init\_\_(self, df, X\_col, y\_col, batch\_size, directory, tokenizer, vocab\_size, max\_length, features,shuffle=True):

        self.df = df.copy()

        self.X\_col = X\_col

        self.y\_col = y\_col

        self.directory = directory

        self.batch\_size = batch\_size

        self.tokenizer = tokenizer

        self.vocab\_size = vocab\_size

        self.max\_length = max\_length

        self.features = features

        self.shuffle = shuffle

        self.n = len(self.df)

    # Hàm được gọi sau mỗi epoch trong quá trình huấn luyện.

    def on\_epoch\_end(self):

        if self.shuffle:

            self.df = self.df.sample(frac=1).reset\_index(drop=True)

    # Hàm trả về số lượng batch trong mỗi epoch.

    def \_\_len\_\_(self):

        return self.n // self.batch\_size

    # Hàm trả về batch dữ liệu tương ứng với index.

    def \_\_getitem\_\_(self,index):

        batch = self.df.iloc[index \* self.batch\_size:(index + 1) \* self.batch\_size,:]

        X1, X2, y = self.\_\_get\_data(batch)

        return (X1, X2), y

    # Hàm xử lý dữ liệu trong một batch.

    def \_\_get\_data(self,batch):

        X1, X2, y = list(), list(), list()

        images = batch[self.X\_col].tolist()

        for image in images:

            feature = self.features[image][0]

            captions = batch.loc[batch[self.X\_col]==image, self.y\_col].tolist()

            for caption in captions:

                seq = self.tokenizer.texts\_to\_sequences([caption])[0]

                for i in range(1,len(seq)):

                    in\_seq, out\_seq = seq[:i], seq[i]

                    in\_seq = pad\_sequences([in\_seq], maxlen=self.max\_length)[0]

                    out\_seq = to\_categorical([out\_seq], num\_classes=self.vocab\_size)[0]

                    X1.append(feature)

                    X2.append(in\_seq)

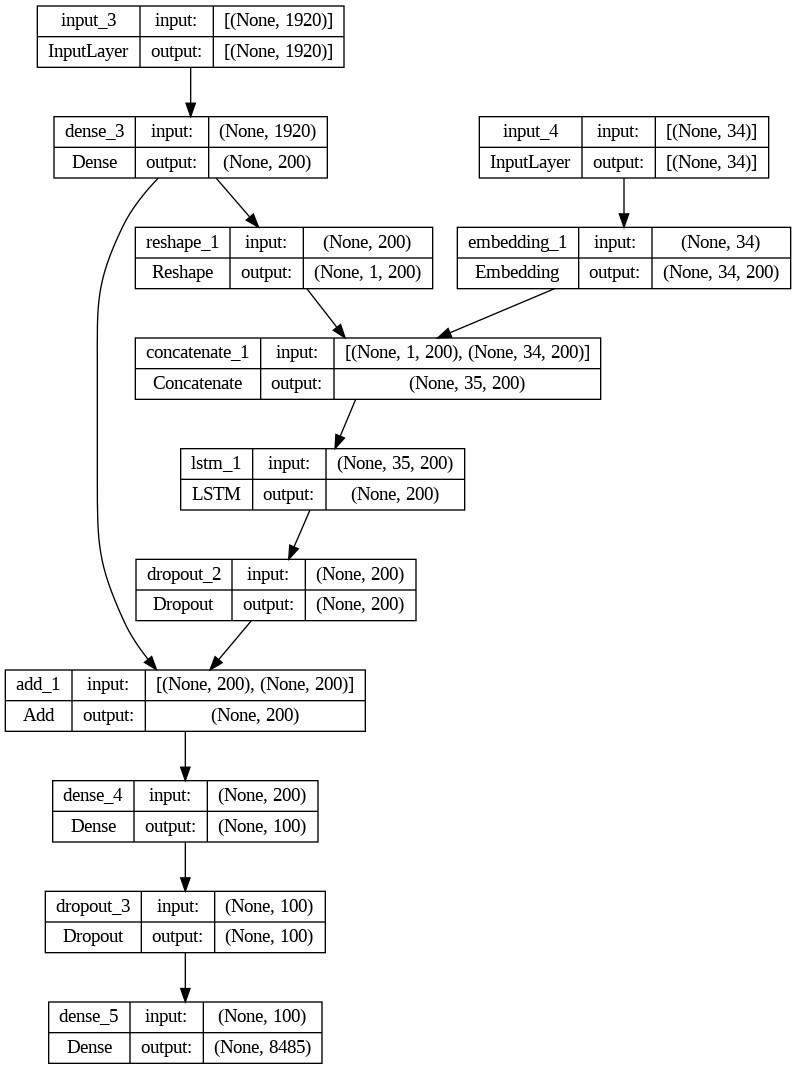
                    y.append(out\_seq)

        X1, X2, y = np.array(X1), np.array(X2), np.array(y)

        return X1, X2, y

## 3.5 Xây dựng mô hình 1

Mô hình sẽ gồm 2 đầu vào là ảnh và từ bắt đầu trong câu. Ảnh sẽ được đưa qua lớp Dense và reshape lại chiều (1, 200). Từ bắt đầu sẽ được đưa qua lớp Embedding để chuyển thành vector đặc trưng do mô hình Glove tạo ra. Sau đó, hai đầu ra sẽ được kết hợp lại với nhau thông qua lớp Concatenate. Đầu ra của lớp Concatenate được đưa qua lớp LSTM và Dropout để học được mối quan hệ của câu. Nhằm để tránh mất mát về dữ liệu, kết quả sẽ được kết hợp với đặc trưng của hình ảnh một lần nữa thông qua lớp Add. Cuối cùng, nó sẽ được đưa qua lớp Fully Connection để dự đoán từ tiếp theo trong câu.



Hình 10. Cấu túc của mô hình 1

input1 = Input(shape=(1920,))

input2 = Input(shape=(max\_length,))

img\_features = Dense(200, activation='relu')(input1)

img\_features\_reshaped = Reshape((1, 200), input\_shape=(200,))(img\_features)

sentence\_features = Embedding(input\_dim=vocab\_size, output\_dim=embedding\_dim, weights=[embedding\_matrix], input\_length=max\_length, trainable=False)(input2)

merged = concatenate([img\_features\_reshaped,sentence\_features],axis=1)

sentence\_features = LSTM(200)(merged)

x = add([sentence\_features, img\_features])

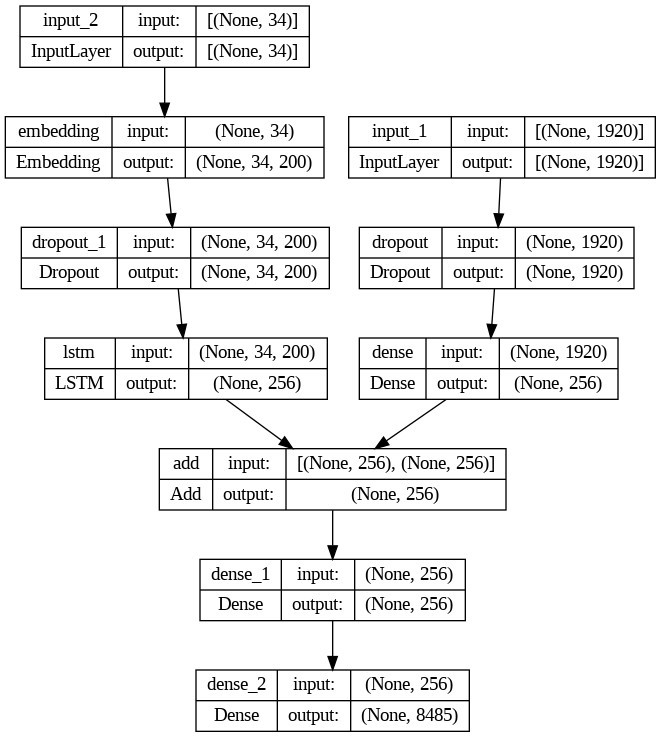
x = Dense(256, activation='relu')(x)

output = Dense(vocab\_size, activation='softmax')(x)

caption\_model = Model(inputs=[input1,input2], outputs=output)

## 3.6 Xây dựng mô hình 2

Trong mô hình 2, ảnh sẽ được trích xuất đặc trưng bằng DenseNet201 và đưa qua lớp Dropout + Dense và biến đổi thành chiều (256). Từ bắt đầu sẽ được đưa qua lớp Embedding để biến đổi thành vector đặc trưng. Vector đặc trưng này do mô hình Glove tạo sẵn. Tiếp theo, từ bắt đầu sẽ được đưa qua lớp LSTM để học được mối quan hệ của câu và biến đổi về chiều (256). Từ sẽ được kết hợp với ảnh có cùng chiều là (256) và sẽ được kết hợp với nhau thông qua lớp Add. Cuối cùng, vector kết hợp sẽ được đưa vào lớp Fully Connected để dự đoán từ tiếp theo trong câu.



Hình 11. Cấu trúc mô hình 2

## 3.7 Đánh giá độ chính xác

input1 = Input(shape=(1920,))

input2 = Input(shape=(max\_length,))

img\_feature1 = Dropout(0.5)(input1)

img\_feature2 = Dense(256, activation='relu')(img\_feature1)

sentence\_feature1 = Embedding(input\_dim=vocab\_size, output\_dim=embedding\_dim, weights=[embedding\_matrix], input\_length=max\_length, trainable=False)(input2

sentence\_feature2 = Dropout(0.5)(sentence\_feature1)

sentence\_feature3 = LSTM(256)(sentence\_feature2)

decoder1 = add([img\_feature2, sentence\_feature3])

decoder2 = Dense(256, activation='relu')(decoder1)

output = Dense(vocab\_size, activation='softmax')(decoder2)

caption\_model = Model(inputs=[input1, input2], outputs=output)

Độ chính xác của mô hình sẽ được đánh giá bằng hàm BLEU. Hàm BLEU sẽ so sánh độ khớp giữa câu dự đoán và câu mẫu. Độ chính xác sẽ nằm trong khoảng từ 0 đến 1, tương ứng với khác nhau hoàn toàn và giống nhau hoàn toàn. Thông thường, độ chính xác ở mức 0.4 trở lên thì mô hình được đánh giá là tốt.

# hàm tạo ra n-gram

def generate\_ngrams(sentence, n):

    words = sentence.lower().split(' ')

    ngrams = []

    for i in range(len(words) - n + 1):

        ngrams.append(" ".join(words[i:i+n]))

    return ngrams

# hàm đánh giá độ khớp của câu mẫu và câu do mô hình dự đoán

def precision\_score(ref\_ngram, can\_ngram):

    if len(can\_ngram) == 0 or len(ref\_ngram) == 0:

        print("Error: câu rỗng")

        return None

    ref\_counts = Counter(ref\_ngram)

    can\_counts = Counter(can\_ngram)

    match\_counts = sum(min(ref\_counts[ngram], can\_counts[ngram]) for ngram in can\_counts)

    p = match\_counts / max(len(can\_ngram), 1)

    return p

def bleu(candidate, references, N=1):

    average = []

    for reference in references:

        lenc = len(candidate.split())

        lenr = len(reference.split())

        BP = 1 if lenc >= lenr else np.exp(1 - (lenr / lenc))

        gram\_score = 0

        for i in range(1, N + 1):

            can\_ngram = generate\_ngrams(candidate, i)

            ref\_ngram = generate\_ngrams(reference, i)

            pn = precision\_score(ref\_ngram, can\_ngram)

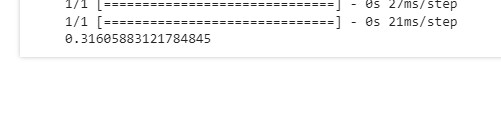
            gram\_score += np.log(pn)

        ref\_score = BP \* np.exp(gram\_score)

        average.append(ref\_score)

    return statistics.mean(average)

Kiểm tra độ chính xác trên tập test đạt 0.316:



Hình 12. Độ chính xác của mô hình theo hàm BLEU

# KẾT LUẬN

**1. Kết quả đạt được**

* + Làm sạch dữ liệu.
  + Xây dựng được mô hình học sâu sinh mô tả cho hình ảnh
  + Hoàn thành báo cáo

**2. Hướng phát triển**

Mặc dù đã hoàn thành được những yêu cầu của đề tài nhưng nhóm chúng em vẫn còn được nhiều thiếu sót như:

* Độ chính xác của mô hình còn thấp.
* Chưa ứng dụng mô hình vào trong thực tiễn.
* Tập dữ liệu vẫn còn nhỏ

Với những thiếu sót nêu trên, hướng phát triển dự kiến là:

* Tìm hiểu các mô hình học sâu tiên tiến hơn
* Thực hiện các chức năng như text to voice,.. để ứng dụng mô hình vào thực tiễn.
* Thu nhập thêm dataset

# DANH MỤC TÀI LIỆU THAM KHẢO

[1] A Guide to Image Captioning (Part 1): Giới thiệu bài toán sinh mô tả cho ảnh của tác giả Nguyễn Thanh Huyền

[2] Mô hình học sâu trên www.kaggle.com

[3] Mô hình học sâu trên github

[4] Sách deep leaning cơ bản