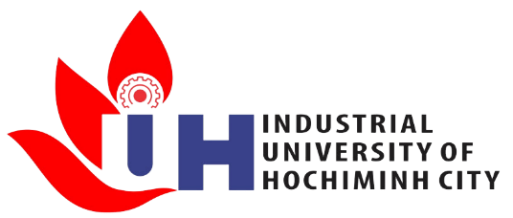
**BỘ CÔNG THƯƠNG**

**TRƯỜNG ĐẠI HỌC CÔNG NGHIỆP TP. HỒ CHÍ MINH**

**KHOA CÔNG NGHỆ THÔNG TIN**



**TRẦN ĐÔNG HOÀNG**

**ỨNG DỤNG DEEP LEARNING TRONG PHÁT SINH MÔ TẢ CHO ẢNH**

**Ngành: Khoa học máy tính**

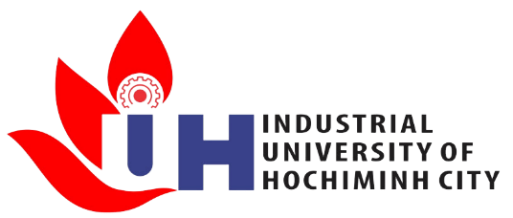
**Giảng viên hướng dẫn: ThS. Võ Quang Hoàng Khang**

**THÀNH PHỐ HỒ CHÍ MINH, THÁNG 12 NĂM 2022**

**BỘ CÔNG THƯƠNG**

**TRƯỜNG ĐẠI HỌC CÔNG NGHIỆP TP. HỒ CHÍ MINH**

**KHOA CÔNG NGHỆ THÔNG TIN**



**TRẦN ĐÔNG HOÀNG**

**ỨNG DỤNG DEEP LEARNING TRONG PHÁT SINH MÔ TẢ CHO ẢNH**

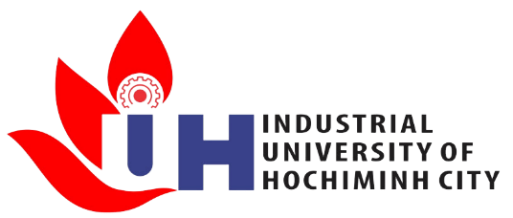
**Ngành: Khoa học máy tính**

**Giảng viên hướng dẫn: ThS. Võ Quang Hoàng Khang**

**THÀNH PHỐ HỒ CHÍ MINH, THÁNG 12 NĂM 2022**

**INDUSTRIAL UNIVERSITY OF HO CHI MINH CITY**

**FACULTY OF INFORMATION TECHNOLOGY**



**TRAN DONG HOANG**

**IMAGE CAPTIONING USING DEEP LEARNING**

**Major: Computer science**

**Instructors: ThS. Vo Quang Hoang Khang**

**HO CHI MINH CITY, DECEMBER 2022**

**ABSTRACT**

Manually assigning image labels for different purposes will take a lot of effort Generating descriptions for images automatically saves users time and costs, and provides better support for users. visually impaired. So, we apply Deep Learning using models VGG16-LSTM, MobileNetV3-Transformer (Attention), VGG16-GRU (Attention) and Flick8k dataset with 8000 images and 5 description sentences to find the best model based on different rating scale

LỜI CẢM ƠN

Khóa luận tốt nghiệp là một dấu mốc lớn với nhóm em, nhóm em rất tự hào sau khi hoàn thành công trình nghiên cứu này về DL. Để hoàn thành tốt được bài khóa luận, nhóm em đã chăm chỉ học hỏi và nỗ lực hết mình và còn nhận được rất nhiều sự giúp đỡ từ gia đình, bạn bè, giáo viên hướng dẫn, bộ môn và nhà trường…

Trước hết, nhóm muốn gửi lời cảm ơn sâu sắc đối với thầy Võ Quang Hoàng Khang. Được làm việc với thầy và được thầy hướng dẫn là những kinh nghiệm quý giá đối với em trong việc hoàn thành khóa luận tốt nghiệp và cả trong sự nghiệp sau này.

Nhóm xin chân thành cảm ơn quý Thầy Cô trong Khoa Cộng Nghệ Thông Tin đã tận tình giảng dạy, trang bị cho chúng em những kiến thức quý báu trong suốt quá trình học tập để có thể thực hiện được đề tài. Và chúng em cũng xin cảm ơn Khoa Cộng Nghệ Thông Tin, trường Đại Học Cộng Nghiệp Tp. Hồ Chí Minh đã tạo điều kiện thuận lợi cho chúng em trong 4 năm học tập tại trường và thực hiện đề tài tốt nghiệp.

Nhóm cũng xin gửi lòng biết ơn đến gia đình và bạn bè, những người đã giúp đỡ, cổ vũ cho các thành viên rất nhiều trong lúc gặp khó khăn cũng như trong suốt quá trình thực hiện thực tập doanh nghiệp.

Mặc dù đã cố gắng hoàn thành thực tập trong phạm vi và khả năng cho phép nhưng chắc chắn sẽ không tránh khỏi những thiếu sót, kính mong nhận được sự góp ý và tận tình chỉ bảo của quý Thầy Cô.

Nhóm em xin chân thành cảm ơn!

TP. Hồ Chí Minh, ngày......tháng......năm 2022

Sinh viên thực hiện

**NHẬN XÉT CỦA GIÁO VIÊN HƯỚNG DẪN** ........................................................................................................................................ ............................................................................................................................................ ............................................................................................................................................ ............................................................................................................................................ ............................................................................................................................................ ............................................................................................................................................ ............................................................................................................................................ ............................................................................................................................................ ............................................................................................................................................ ............................................................................................................................................ ............................................................................................................................................ ............................................................................................................................................ ............................................................................................................................................ ............................................................................................................................................ ............................................................................................................................................ ............................................................................................................................................ ............................................................................................................................................ ............................................................................................................................................ ............................................................................................................................................ ............................................................................................................................................ ............................................................................................................................................ ............................................................................................................................................

TP. Hồ Chí Minh, ngày......tháng......năm 2022

Sinh viên thực hiện

**NHẬN XÉT CỦA GIÁO VIÊN PHẢN BIỆN 1** ........................................................................................................................................ ............................................................................................................................................ ............................................................................................................................................ ............................................................................................................................................ ............................................................................................................................................ ............................................................................................................................................ ............................................................................................................................................ ............................................................................................................................................ ............................................................................................................................................ ............................................................................................................................................ ............................................................................................................................................ ............................................................................................................................................ ............................................................................................................................................ ............................................................................................................................................ ............................................................................................................................................ ............................................................................................................................................ ............................................................................................................................................ ............................................................................................................................................ ............................................................................................................................................ ............................................................................................................................................ ............................................................................................................................................ ............................................................................................................................................

TP. Hồ Chí Minh, ngày......tháng......năm 2022

Sinh viên thực hiện

**NHẬN XÉT CỦA GIÁO VIÊN PHẢN BIỆN 2** ........................................................................................................................................ ............................................................................................................................................ ............................................................................................................................................ ............................................................................................................................................ ............................................................................................................................................ ............................................................................................................................................ ............................................................................................................................................ ............................................................................................................................................ ............................................................................................................................................ ............................................................................................................................................ ............................................................................................................................................ ............................................................................................................................................ ............................................................................................................................................ ............................................................................................................................................ ............................................................................................................................................ ............................................................................................................................................ ............................................................................................................................................ ............................................................................................................................................ ............................................................................................................................................ ............................................................................................................................................ ............................................................................................................................................ ............................................................................................................................................

TP. Hồ Chí Minh, ngày......tháng......năm 2022

Sinh viên thực hiện

**MỤC LỤC**

[**DANH MỤC HÌNH ẢNH** 11](#_Toc121371994)

[**DANH MỤC CÁC THUẬT NGỮ VIẾT TẮT** 12](#_Toc121371995)

[**CHƯƠNG 1: TỔNG QUÁT** 14](#_Toc121371996)

[**1.1** **Giới thiệu** 14](#_Toc121371997)

[**1.2** **Mục tiêu nghiên cứu** 14](#_Toc121371998)

[**1.3** **Nội dung nghiên cứu** 15](#_Toc121371999)

[**1.4** **Đối tượng và phạm vi nghiên cứu** 15](#_Toc121372000)

[**1.5** **Phương pháp nghiên cứu** 15](#_Toc121372001)

[**CHƯƠNG 2: CƠ SỞ LÝ THUYẾT** 16](#_Toc121372002)

[**2.1** **Giới thiệu về học sâu** 16](#_Toc121372004)

[**2.2** **Convolutional Neural Network (CNN)** 18](#_Toc121372005)

[**2.2.1** **Giới thiệu** 19](#_Toc121372006)

[**2.2.2** **ConvNets thay cho Feed-Forward Neural Nets** 20](#_Toc121372007)

[**2.2.3** **Hình ảnh đầu vào** 21](#_Toc121372008)

[**2.2.4** **Convolution Layer — The Kernel** 21](#_Toc121372009)

[**2.2.5** **Pooling Layer** 27](#_Toc121372010)

[**2.2.6** **Phân loại — Lớp kết nối đầy đủ (Lớp FC)** 29](#_Toc121372011)

[**2.3** **VGG-16** 30](#_Toc121372012)

[**2.3.1** **Kiến trúc VGG :** 31](#_Toc121372013)

[**2.3.2** **Cấu hình:** 33](#_Toc121372014)

[**2.3.3** **Kết quả:** 33](#_Toc121372015)

[**2.3.4** **Hạn chế của VGG 16:** 34](#_Toc121372016)

[**2.4** **MobileNetV3** 34](#_Toc121372017)

[**2.4.1** **Mô hình Squeeze and Excitation (SE)** 34](#_Toc121372018)

[**2.4.2** **Kiến trúc mạng** 34](#_Toc121372019)

[**2.5** **Recurrent neural network (RNN)** 36](#_Toc121372020)

[**2.5.1** **Kiến trúc của một mạng RNN truyền thống** 36](#_Toc121372021)

[**2.5.2** **Ứng dụng của RNNs** 38](#_Toc121372022)

[**2.5.3** **Hàm mất mát** 40](#_Toc121372023)

[**2.5.4** **Lan truyền ngược theo thời gian** 40](#_Toc121372024)

[**2.5.5** **Các hàm kích hoạt thường dùng** 40](#_Toc121372025)

[**2.5.6** **Vanishing/exploding gradient** 41](#_Toc121372026)

[**2.5.7** **Gradient clipping** 41](#_Toc121372027)

[**2.5.8** **Các loại cổng** 42](#_Toc121372028)

[**2.5.9** **GRU/LSTM** 43](#_Toc121372029)

[**2.5.10** **Giải thích và các kí hiệu** 44](#_Toc121372030)

[**2.5.11** **Embedding matrix** 45](#_Toc121372031)

[**2.5.12** **Word embeddings GloVe** 45](#_Toc121372032)

[**2.5.13** **Điểm Bleu** 46](#_Toc121372033)

[**2.5.14** **Attention model** 46](#_Toc121372034)

[**2.6** **Long short term memory (LSTM)** 48](#_Toc121372035)

[**2.6.1** **Mô hình LSTM** 48](#_Toc121372036)

[**2.6.2** **LSTM chống vanishing gradient** 50](#_Toc121372037)

[**2.7** **Transformers** 51](#_Toc121372038)

[**2.7.1** **Tại sao Transformers lại quan trọng** 53](#_Toc121372039)

[**CHƯƠNG 3: KẾT QUẢ VÀ THỰC NGHIỆM** 54](#_Toc121372040)

[**3.1** **Chuẩn bị dữ liệu cho huấn luyện** 54](#_Toc121372042)

[**3.2** **Kết quả của các mô hình đã sử dụng** 55](#_Toc121372043)

[**3.2.1** **VGG16-LSTM** 55](#_Toc121372044)

[**3.2.2** **MobileNetV3 – Transformer (Attention)** 57](#_Toc121372045)

[**3.2.3** **VGG16- GRU (Attention)** 58](#_Toc121372046)

[**3.3.** **So sánh độ chính xác của các mô hình.** 58](#_Toc121372052)

[**3.4.** **Dự đoán mô hình** 58](#_Toc121372053)

[**3.4.1.** **VGG16-LSTM** 58](#_Toc121372054)

[**3.4.2.** MobileNetV3 – Transformer (Attention) 60](#_Toc121372063)

[**3.4.3.** **VGG16-GRU (Attention)** 61](#_Toc121372064)

[**CHƯƠNG 4: KẾT LUẬN VÀ HƯỚNG PHÁT TRIỂN** 64](#_Toc121372065)

[**4.1** **Kết luận** 64](#_Toc121372067)

[**4.2** **Hạn chế** 65](#_Toc121372068)

[**4.3** **Hướng phát triển** 65](#_Toc121372069)

[**TÀI LIỆU THAM KHẢO** 65](#_Toc121372070)

# 

# **DANH MỤC HÌNH ẢNH**

[Hình 1: Mô hình CNN 15](#_Toc120500729)

[Hình 2: Trình tự CNN để phân loại các chữ số viết tay 16](#_Toc120500730)

[Hình 3: Làm phẳng ma trận ảnh 3x3 thành vectơ 9x1 17](#_Toc120500731)

[Hình 4: Hình ảnh RGB 4x4x3 18](#_Toc120500732)

[Hình 5: Kết hợp hình ảnh 5x5x1 với nhân 3x3x1 để có tính năng tích hợp 3x3x1 19](#_Toc120500733)

[Hình 6: Chuyển động của hạt nhân 20](#_Toc120500734)

[Hình 7: Thao tác tích chập trên ma trận hình ảnh MxNx3 với Hạt nhân 3x3x3 21](#_Toc120500735)

[Hình 8: Thao tác tích chập với độ dài stride = 2 22](#_Toc120500736)

[Hình 9: CÙNG Phần đệm: hình ảnh 5x5x1 được đệm bằng 0 để tạo hình ảnh 6x6x1 23](#_Toc120500737)

[Hình 10: Pooling 3x3 trên tính năng tích hợp 5x5 24](#_Toc120500738)

[Hình 11: Các loại gộp 25](#_Toc120500739)

[Hình 12: Fully Connected Layer 26](#_Toc120500740)

[Hình 13: Kiến trúc VGG-16 28](#_Toc120500741)

[Hình 14: Bản đồ kiến trúc VGG-16 29](#_Toc120500742)

[Hình 15: Các cấu hình VGG khác nhau 30](#_Toc120500743)

[Hình 16: Mô hình LSTM 45](#_Toc120500744)

[Hình 17: cell state trong LSTM 46](#_Toc120500745)

[Hình 18: Ứng dụng Transformer vào dịch máy. Nguồn: Google AI Blog 48](#_Toc120500746)

[Hình 19: Phân phối tự chú ý của bộ mã hóa cho từ “nó” từ lớp thứ 5 đến lớp thứ 6 của Máy biến áp được đào tạo về bản dịch tiếng Anh sang tiếng Pháp (một trong tám đầu chú ý). Nguồn: Google AI Blog . 50](#_Toc120500747)

[Hình 20: Tập dữ liệu và mô tả ảnh 51](#_Toc120500748)

[Hình 21: Đồ thị hàm Loss trên mô hình VGG16-LSTM 52](#_Toc120500749)

[Hình 22: Đồ thị độ chính xác trên mô hình VGG16-LSTM 52](#_Toc120500750)

[Hình 23: Đồ thị hàm Loss trên mô hình MobileNetV3-Transformer 53](#_Toc120500751)

[Hình 24: Đồ thị độ chính xác trên mô hình MobileNetV3 – Transformer 53](#_Toc120500752)

[Hình 25: Đồ thị hàm Loss trên mô hình VGG16-GRU 54](#_Toc120500753)

[Hình 26: Kết quả dự đoán mô hình VGG16-LSTM 55](#_Toc120500754)

[Hình 27: Kết quả dự đoán mô hình MobileNetV3 – Transformer (Attention) 56](#_Toc120500755)

[Hình 28: Kết quả dự đoán mô hình MobileNetV3 – Transformer (Attention) 57](#_Toc120500756)

[Hình 29: Kết quả dự đoán mô hình VGG16-GRU 57](#_Toc120500757)

[Hình 30: Kết quả dự đoán mô hình VGG16-GRU (Attention) 58](#_Toc120500758)

[Hình 31: Kết quả dự đoán mô hình VGG16-GRU 59](#_Toc120500759)

[Hình 32: Kết quả dự đoán mô hình VGG16-GRU (Attention) 60](#_Toc120500760)

# **DANH MỤC CÁC THUẬT NGỮ VIẾT TẮT**

|  |  |  |
| --- | --- | --- |
| Từ viết tắt | Từ đầy đủ | Nghĩa |
| ANN | Artifical neural network | Mạng thần kinh nhân tạo |
| CNN | Convolution neural network | Mạng nơron tích chập |
| RNN | Recurrent neural network | Mạng nơron hồi quy |
| AI | Artificial intelligence | Trí tuệ nhân tạo |
| DL | Deep learning | Học sâu |
| CONV | Convolutional layer | Lớp tích chập |
| FC | Fully connected layer | Lớp kết nối đầy đủ |
| GPU | Graphics Processing Unit | Đơn vị xử lý đồ hoạ |

# **CHƯƠNG 1: TỔNG QUÁT**

## **Giới thiệu**

Hiện nay, chúng ta đang sống trong thời đại phát triển mạnh mẽ của công nghệ. Trí tuệ nhân tạo (Artificial Intelligence), Dữ liệu lớn ( Big data),.. những công nghệ tương lai là một ngành thuộc lĩnh vực khoa học máy tính (Computer Science). Trong đó thị giác máy tính đã có nhiều bước tiến lớn, phát triển, tiến bộ đáng kể về nhận dạng hình ảnh, nhận dạng đối tượng điển hình là những ứng dụng các kĩ thuật học máy (Machine Learning), học sâu (Deep Learning) vào trong thị giác máy tính. Các mô hình kinh điển, làm nền tảng cho như mạng neural tái tạo (Recurrent neural networks - RNN), mạng neural nhân tạo (ANN), mạng neural tích chập (Convolutional neural networks – CNN). Trong đó CNN được xem như là một bước tiến lớn giúp cải thiện độ chính xác khi tích hợp một lớp vào, một lớp ra và nhiều lớp ẩn khác nhau. Ngoài ra còn đóng vai trò lớn trong việc phát triển ra những mô hình khác như VGG16, Resnet, Inception, …Bên cạnh hỗ trợ cải thiện độ chính xác và tốc độ, các mô hình đó còn giúp hỗ trợ nhận dạng đối tượng realtime trên các ứng dụng thực tế một cách tiện lợi, nhỏ gọn với các thiết bị như nhận dạng biển báo giao thông hay các thiết bị y khoa cần độ chính xác rất cao. Một số những mô hình nổi bật được sử dụng để giải quyết những vấn đề phức tạp cao như RCNN, YOLO, Unet, Inceptionv3. Ứng dụng cho bài toán gán nhãn hình ảnh rõ ràng nhất đó chính là phát sinh mô tả cho ảnh. Với số lượng ảnh càng ngày càng khổng lồ với nguồn dữ liệu Big data việc gán nhãn hình ảnh để được sử dụng vào các mục đích khác nhau một cách thủ công sẽ tiêu tốn rất nhiều thời gian, tiền bạc và công sức. Vì vậy ứng dụng phát sinh mô tả cho ảnh giúp tự động hóa cải thiện tốc độ và độ chính xác cho ảnh. Ngoài ra còn ứng dụng hỗ trợ tốt cho những người khiếm thị.

## **Mục tiêu nghiên cứu**

Để nghiên cứu và cho ra ứng dụng phát sinh nhãn cho hình ảnh cần được đảm bảo độ chính xác cao và tốc độ nhanh, chính vì thế nhóm phải thiết kế mô hình và thử qua nhiều lần tối ưu. So sánh các mô hình như VGG16, MobileNetV3, LSTM, Transformer, GRU, Attention để cải tiến và tìm ra một mô hình tốt nhất dựa vào độ chính xác và kết quả thực nghiệm kiểm thử. Chính vì thế mục tiêu nghiên cứu của nhóm sẽ là thiết kế cải tiến mô hình, sau đó so sánh với các mô hình tốt hiện nay và cho ra mô hình hoàn chỉnh nhất đảm bảo được độ chính xác và tốc độ của bài toán.

## **Nội dung nghiên cứu**

Để hoàn thành mục tiêu được đề ra thì cần phải thực hiện một số nội dung sau:

* Thu thập dữ liệu, gán nhãn cho dữ liệu.
* Nghiên cứu, thiết kế mô hình máy học sử dụng trong việc gán nhãn ảnh.
* Nghiên cứu các mô hình có sẵn thử với dữ liệu thực.
* Nghiên cứu cách cải tiến, tối ưu mô hình .

## **Đối tượng và phạm vi nghiên cứu**

Đối tượng: hình ảnh được gán nhãn .

Phạm vi nghiên cứu:

* Tổng quan về phát sinh mô tả cho ảnh.
* Tổng quan về Deep Learning
* Nghiên cứu các khái niệm cơ bản về mạng CNN, RNN
* Nghiên cứu các mô hình như VGG16, MobileNetV3, LSTM, Transformer, GRU, Attention.
* Đánh giá độ chính xác và kết quả thực nghiệm tìm ra mô hình tốt nhất.

## **Phương pháp nghiên cứu**

* Thu thập dữ liệu ảnh trên website, dữ liệu từ các bài báo uy tín.
* Nguyên cứu các mô hình học sâu VGG16, MobileNetV3, LSTM, Transformer, GRU, Attention dựa vào đó có thể điều chỉnh số lượng tham số trong mỗi mô hình khi đưa vào huấn luyện.
* Đưa ra so sánh và tối ưu lại mô hình đưa ra độ chính xác và tốc độ tốt nhất.

# **CHƯƠNG 2: CƠ SỞ LÝ THUYẾT**

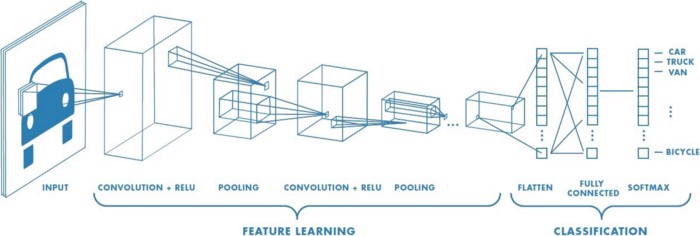


## Giới thiệu về học sâu

Từ khi các chương trình máy tính được tạo ra, con người luôn mong muốn tạo ra các chương trình có trí tuệ giống như con người, chẳng hạn như có thể hiểu được nội dung của ảnh, tiếng nói, tự động làm những công việc thường ngày của con người, ... Hướng nghiên cứu các thuật toán mô phỏng trí tuệ của con người được gọi chung là *trí tuệ nhân tạo (artificial intelligence - AI)*. Một trong những thách thức của *trí tuệ nhân tạo* đó là giải những bài toán tưởng chừng như đơn giản với con người nhưng lại thực sự rất khó giải quyết bằng máy tính, bởi vì bài toán này không thể mô tả dưới dạng hình thức; chẳng hạn nhận diện mặt người trong ảnh, nhận diện từ trong tiếng nói, ... Để giải quyết các thách thức này, một số nghiên cứu đã cố gắng biểu diễn lại tri thức của thế giới xung quanh ta (ngôn ngữ tự nhiên, hình ảnh, ...) thành dưới dạng mô tả hình thức bằng cách sử dụng các luật suy diễn. Hướng tiếp cận này được gọi là *hệ cơ sở tri thức (knowledge base)*. Tuy nhiên, khó khăn lớn nhất của hướng tiếp cận này đó làcác luật suy diễn mà con người tạo ra không đủ phức tạp để mô tả được các tri thức của thế giới quanh ta. Do đó, một số nghiên cứu đã đề xuất thay vì con người tự thiết kế các luật suy diễn để mô tả tri thức cho máy tính hiểu, thì các hệ thống AI tự học các tri thức bằng cách rút trích các mẫu (pattern) từ dữ liệu ban đầu. Hướng tiếp cận này được gọi là *máy học (machine learning)*. Học sâu là một lĩnh vực của máy học liên quan đến các thuật toán lấy cảm hứng từ cấu trúc và chức năng của bộ não được gọi là mạng lưới thần kinh nhân tạo. Nhóm tác giả Yann LeCunn, Yoshua Bengio, Geoffey Hinton đã đề cập đến học sâu trong một nghiên cứu tại tạp chí Nature vào năm 2015 [5]. *“Học sâu cho phép các mô hình tính toán sử dụng nhiều tầng (lớp) xử lý để học được cách biểu diễn lại dữ liệu với nhiều lớp trừu tượng. Các phương pháp này cải thiện mạnh mẽ và đạt được kết quả rất tốt trong các bài toán như nhận dạng giọng nói, phát hiện và nhận dạng vật thể, cùng rất nhiều ứng dụng trong các lĩnh vực khác như khám phá các loại thuốc, nghiên cứu về gen. Học sâu phát hiện ra cấu trúc phức tạp trong tập dữ liệu lớn bằng cách sử dụng thuật toán lan truyền ngược (backpropagation) để chỉ ra cách mà một máy tính có thể thay đổi các trọng số trung gian giữa các lớp. Mạng nơ-ron tích chập có rất nhiều đột phá trong xử lý ảnh, xử lý video, giọng nói, audio, ... trong khi đó mạng nơ-ron hồi quy lại ứng dụng trong xử lý dữ liệu dạng chuỗi như như văn bản, giọng nói”*.

Những năm gần đây học sâu đã tạo ra những bước đột phá đáng kể trong lĩnh vực trí tuệ nhân tạo. Các mô hình học sâu dần trở thành các mô hình tiên tiến, giúp giải quyết hiệu quả nhiều bài toán về thị giác máy tính, nhận dạng tiếng nói, xử lý ngôn ngữ tự nhiên, ... có thể kể đến như dịch máy, phân tích ngữ nghĩa, chatbot, bài toán hỏi đáp, bài toán đọc hiểu, mô tả ảnh. Ở những phần tiếp theo, chúng tôi sẽ trình bày ba mô hình học sâu được sử dụng phổ biến hiện nay, đó là mô hình mạng nơ-ron tích chập (Convolutional Neural Network) - một mô hình tiên tiến được sử dụng cho các bài toán thị giác máy tính, mô hình mạng nơ-ron hồi quy (Recurrent Neural Network) một mô hình đạt được nhiều bước đột phá trong lĩnh vực xử lý ngôn ngữ tự nhiên, và mô hình bộ nhớ dài-ngắn (Long Short-Term Memory) - mô hình cải tiến của RNN để giải quyết bài toán tự động mô tả nội dung ảnh

## **Convolutional Neural Network (CNN)**

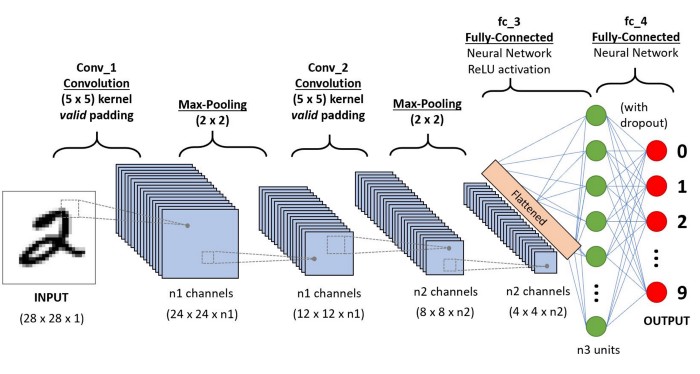


Hình 1: Mô hình CNN

Trí tuệ nhân tạo đã và đang chứng kiến ​​sự phát triển vượt bậc trong việc thu hẹp khoảng cách giữa khả năng của con người và máy móc. Các nhà nghiên cứu cũng như những người đam mê, làm việc trên nhiều khía cạnh của lĩnh vực này để biến những điều tuyệt vời thành hiện thực. Một trong nhiều lĩnh vực như vậy là lĩnh vực Thị giác máy tính.

Mục tiêu của lĩnh vực này là cho phép máy móc nhìn thế giới giống như con người, nhận thức thế giới theo cách tương tự và thậm chí sử dụng kiến ​​thức cho vô số tác vụ như nhận dạng Hình ảnh & Video, Phân tích & Phân loại Hình ảnh, Giải trí Phương tiện, Đề xuất Hệ thống, Xử lý ngôn ngữ tự nhiên, v.v. Những tiến bộ trong Thị giác máy tính với Học sâu đã được xây dựng và hoàn thiện theo thời gian, chủ yếu dựa trên một thuật toán cụ thể - **Convolutional Neural Network.**

### **Giới thiệu**

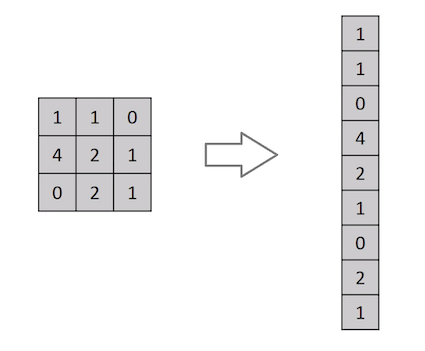


Hình 2: Trình tự CNN để phân loại các chữ số viết tay

**Convolutional Neural Network (ConvNet/CNN)** là một thuật toán Deep Learning có thể lấy hình ảnh đầu vào, gán tầm quan trọng (trọng số và độ lệch có thể học được) cho các khía cạnh/đối tượng khác nhau trong hình ảnh và có thể phân biệt cái này với cái kia. Yêu cầu tiền xử lý trong ConvNet thấp hơn nhiều so với các thuật toán phân loại khác. Mặc dù ở các phương thức nguyên thủy, các bộ lọc được thiết kế thủ công, nhưng nếu được đào tạo đầy đủ, ConvNets có khả năng tìm hiểu các bộ lọc/đặc điểm này.

Kiến trúc của ConvNet tương tự như kiến ​​trúc của mô hình kết nối của các Tế bào thần kinh trong Bộ não con người và được lấy cảm hứng từ cách tổ chức của Visual Cortex. Các tế bào thần kinh riêng lẻ chỉ phản ứng với các kích thích trong một vùng hạn chế của trường thị giác được gọi là Trường tiếp nhận. Một tập hợp các trường như vậy chồng lên nhau để bao phủ toàn bộ khu vực trực quan.

### **ConvNets thay cho Feed-Forward Neural Nets**

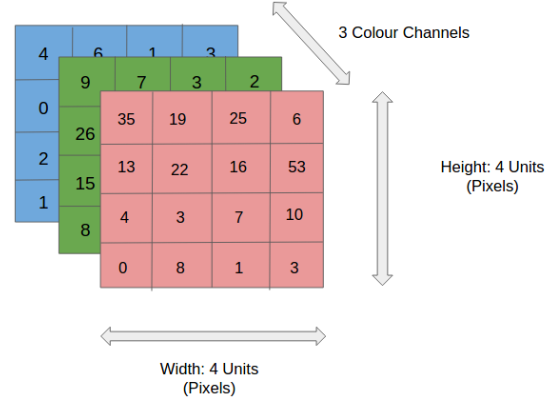


Hình 3: Làm phẳng ma trận ảnh 3x3 thành vectơ 9x1

Trong trường hợp hình ảnh nhị phân cực kỳ cơ bản, phương pháp này có thể hiển thị điểm chính xác trung bình khi thực hiện dự đoán các lớp nhưng sẽ có rất ít hoặc không có độ chính xác khi nói đến hình ảnh phức tạp có các pixel phụ thuộc xuyên suốt.

ConvNet có thể **nắm bắt thành công các yếu tố phụ thuộc Không gian và Thời gian** trong một hình ảnh thông qua việc áp dụng các bộ lọc có liên quan. Kiến trúc thực hiện phù hợp hơn với tập dữ liệu hình ảnh do giảm số lượng tham số liên quan và khả năng sử dụng lại trọng số. Nói cách khác, mạng có thể được đào tạo để hiểu rõ hơn về độ tinh xảo của hình ảnh.

### **Hình ảnh đầu vào**

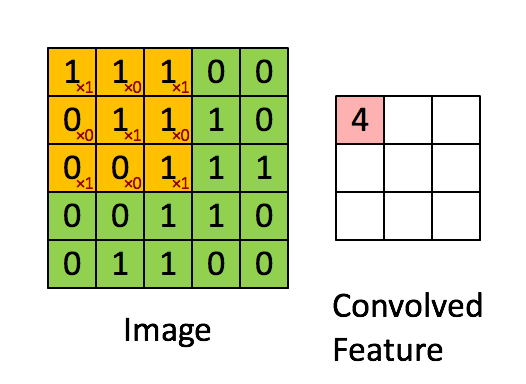


Hình 4: Hình ảnh RGB 4x4x3

Trong hình, chúng ta có một hình ảnh RGB được phân tách bằng ba mặt phẳng màu — Đỏ, Lục và Lam. Có một số không gian màu như vậy trong đó hình ảnh tồn tại — Thang độ xám, RGB, HSV, CMYK, v.v.

Bạn có thể tưởng tượng mức độ phức tạp của tính toán khi hình ảnh đạt đến kích thước, chẳng hạn như 8K (7680×4320). Vai trò của ConvNet là giảm hình ảnh thành một dạng dễ xử lý hơn mà không làm mất đi các tính năng quan trọng để có được dự đoán tốt. Điều này rất quan trọng khi chúng ta thiết kế một kiến ​​trúc không chỉ học tốt các tính năng mà còn có thể mở rộng thành các tập dữ liệu lớn.

### **Convolution Layer — The Kernel**



Hình 5: Kết hợp hình ảnh 5x5x1 với nhân 3x3x1 để có tính năng tích hợp 3x3x1

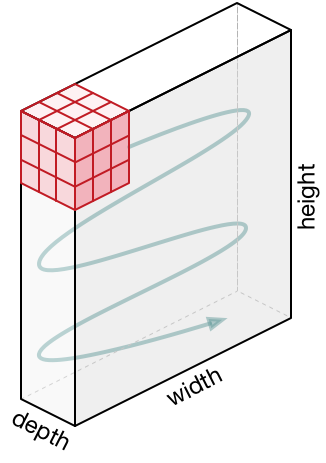
Kích thước hình ảnh = 5 (Chiều cao) x 5 (Chiều rộng) x 1 (Số kênh, ví dụ: RGB)

Trong phần trình diễn ở trên, phần màu xanh lục giống với **hình ảnh đầu vào 5x5x1 của chúng tôi, I** . Phần tử liên quan đến hoạt động tích chập trong phần đầu tiên của Lớp tích chập được gọi là **Hạt nhân/Bộ lọc, K** , được thể hiện bằng màu vàng. Chúng tôi đã chọn **K là ma trận 3x3x1.**

Hạt nhân/Bộ lọc, K =

1 0 1   
0 1 0   
1 0 1

Hạt nhân dịch chuyển 9 lần do **Độ dài Strided = 1 (Non-Strided)**, mỗi lần thực hiện **thao tác nhân theo phần tử giữa K và phần P của hình ảnh** mà hạt nhân đang lơ lửng trên đó.



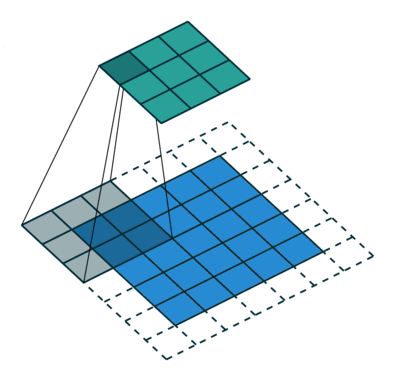
Hình 6: Chuyển động của hạt nhân

Bộ lọc di chuyển sang phải với một Giá trị Stride nhất định cho đến khi nó phân tích cú pháp toàn bộ chiều rộng. Tiếp tục, nó nhảy xuống phần đầu (bên trái) của hình ảnh với cùng Giá trị Stride và lặp lại quy trình cho đến khi toàn bộ hình ảnh được duyệt qua.



Hình 7: Thao tác tích chập trên ma trận hình ảnh MxNx3 với Hạt nhân 3x3x3

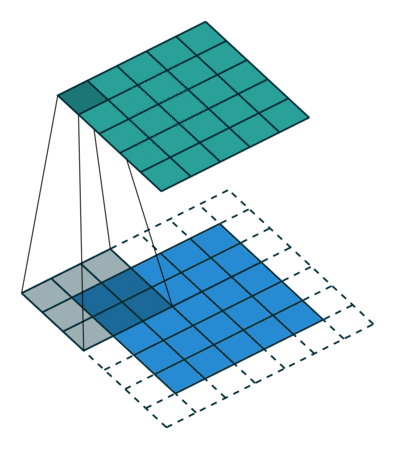
Trong trường hợp hình ảnh có nhiều kênh (ví dụ: RGB), Kernel có cùng độ sâu với hình ảnh đầu vào. Phép nhân ma trận được thực hiện giữa Kn và Trong ngăn xếp ([K1, I1]; [K2, I2]; [K3, I3]) và tất cả các kết quả được tính tổng với độ lệch để cung cấp cho chúng ta Đầu ra tính năng phức tạp của kênh một độ sâu.



Hình 8: Thao tác tích chập với độ dài stride = 2

Mục tiêu của Convolution Operation là **trích xuất các tính năng cấp cao** như các cạnh, từ hình ảnh đầu vào. ConvNets không cần chỉ giới hạn ở một Lớp kết hợp. Thông thường, ConvLayer đầu tiên chịu trách nhiệm nắm bắt các tính năng Cấp thấp như cạnh, màu sắc, hướng chuyển màu, v.v. Với các lớp được thêm vào, kiến ​​trúc cũng thích ứng với các tính năng Cấp cao, mang lại cho chúng tôi một mạng có hiểu biết tốt. hình ảnh trong bộ dữ liệu, tương tự như cách chúng tôi làm.

Có hai loại kết quả đối với thao tác — một loại trong đó tính năng được tích hợp bị giảm kích thước so với đầu vào và loại còn lại trong đó kích thước được tăng lên hoặc giữ nguyên. Điều này được thực hiện bằng cách áp dụng Phần **Valid Padding** trong trường hợp phần trước hoặc **Same Padding** trong trường hợp phần đệm sau.

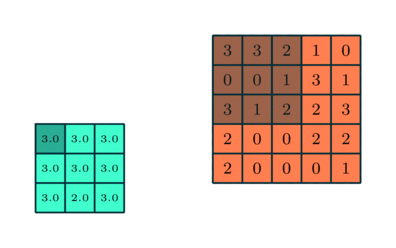


Hình 9: CÙNG Phần đệm: hình ảnh 5x5x1 được đệm bằng 0 để tạo hình ảnh 6x6x1

Khi chúng tôi tăng cường hình ảnh 5x5x1 thành hình ảnh 6x6x1 và sau đó áp dụng nhân 3x3x1 lên trên nó, chúng tôi thấy rằng ma trận tích chập hóa ra có kích thước 5x5x1. Do đó có tên — **Same Padding** .

Mặt khác, nếu chúng ta thực hiện cùng một thao tác mà không có phần đệm, thì chúng ta sẽ thấy một ma trận có kích thước của chính Hạt nhân (3x3x1) - **Valid Padding**.

### **Pooling Layer**

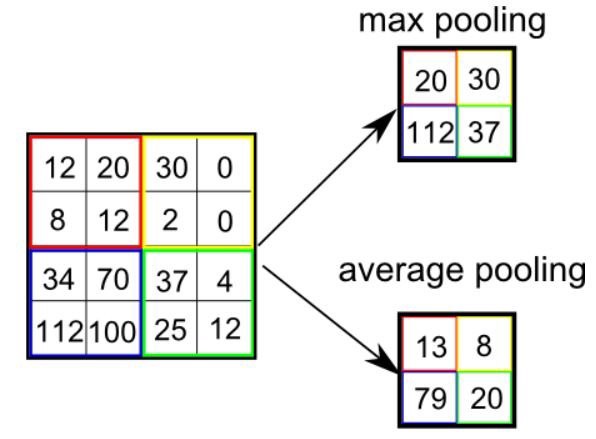


Hình 10: Pooling 3x3 trên tính năng tích hợp 5x5

Tương tự như Convolutional Layer, Pooling layer chịu trách nhiệm giảm kích thước không gian của Convolved Feature. Điều này là để **giảm sức mạnh tính toán cần thiết để xử lý dữ liệu** thông qua việc giảm kích thước. Hơn nữa, nó rất hữu ích để **trích xuất các tính năng vượt trội** là bất biến xoay và vị trí, do đó duy trì quá trình đào tạo mô hình một cách hiệu quả.

Có hai loại Pooling: Max Pooling và Average Pooling. **Max Pooling** trả về **giá trị tối đa** từ phần hình ảnh được Kernel bao phủ. Mặt khác, **Average Pooling** trả về **giá trị trung bình của tất cả các giá trị** từ phần hình ảnh được bao phủ bởi Hạt nhân.

Max Pooling cũng hoạt động như một chất **Noise Suppressant**. Nó loại bỏ hoàn toàn các kích hoạt ồn ào và cũng thực hiện khử nhiễu cùng với giảm kích thước. Mặt khác, Average Pooling chỉ đơn giản thực hiện giảm kích thước như một cơ chế khử tiếng ồn. Do đó, chúng ta có thể nói rằng **Max Pooling hoạt động tốt hơn nhiều so với Average Pooling**.

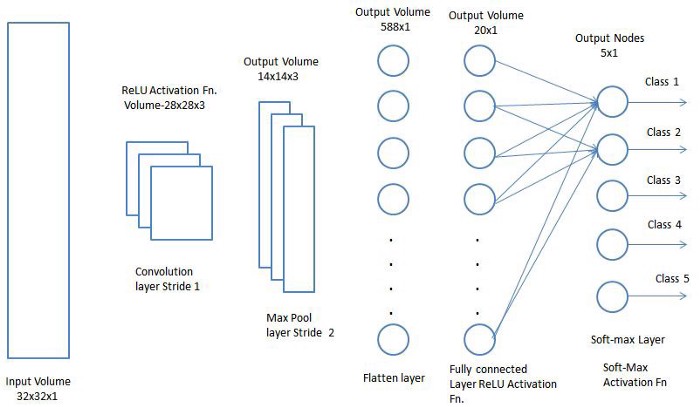


Hình 11: Các loại gộp

Lớp Tích chập và Lớp Tổng hợp, cùng nhau tạo thành lớp thứ i của Mạng Nơ-ron Tích chập. Tùy thuộc vào độ phức tạp của hình ảnh, số lượng các lớp như vậy có thể tăng lên để thu được các chi tiết ở mức độ thấp hơn nữa, nhưng phải trả giá bằng sức mạnh tính toán nhiều hơn.

Sau khi trải qua quá trình trên, chúng tôi đã kích hoạt thành công mô hình để hiểu các tính năng. Tiếp tục, chúng ta sẽ làm phẳng đầu ra cuối cùng và đưa nó vào Mạng nơ-ron thông thường cho mục đích phân loại.

### **Phân loại — Lớp kết nối đầy đủ (Lớp FC)**



Hình 12: Fully Connected Layer

Thêm một lớp Kết nối đầy đủ là một cách (thường) rẻ tiền để học các kết hợp phi tuyến tính của các tính năng cấp cao như được biểu thị bằng đầu ra của lớp tích chập. Lớp Kết nối đầy đủ đang học một hàm có thể phi tuyến tính trong không gian đó.

Bây giờ chúng tôi đã chuyển đổi hình ảnh đầu vào của mình thành một dạng phù hợp cho Perceptron đa cấp độ của chúng tôi, chúng tôi sẽ làm phẳng hình ảnh thành một vectơ cột. Đầu ra phẳng được đưa vào mạng thần kinh chuyển tiếp nguồn cấp dữ liệu và lan truyền ngược được áp dụng cho mỗi lần lặp đào tạo. Trong một loạt các kỷ nguyên, mô hình có thể phân biệt giữa các tính năng nổi trội và một số tính năng cấp thấp nhất định trong hình ảnh và phân loại chúng bằng kỹ thuật **Phân loại Softmax** .

Có sẵn nhiều kiến ​​trúc CNN khác nhau, vốn là chìa khóa trong việc xây dựng các thuật toán cung cấp năng lượng và sẽ cung cấp năng lượng cho toàn bộ AI trong tương lai gần. Một số trong số đó đã được liệt kê dưới đây:

LeNet

AlexNet

VGGNet

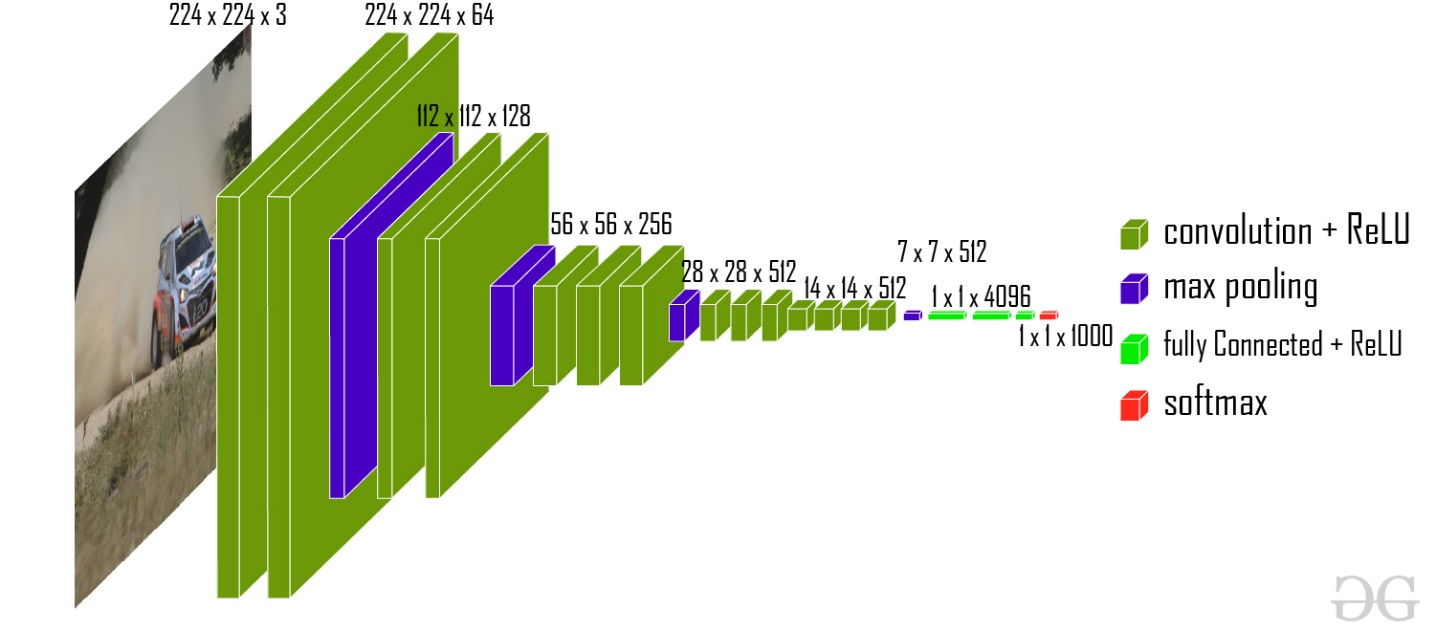
GoogleLeNet

ResNet

ZFNet

## **VGG-16**

**Thử thách nhận dạng hình ảnh quy mô lớn** ImageNet (ILSVRC) là cuộc thi thị giác máy tính hàng năm. Mỗi năm, các đội cạnh tranh trên hai nhiệm vụ. Đầu tiên là phát hiện các đối tượng trong một hình ảnh đến từ *200* lớp, được gọi là bản địa hóa đối tượng. Thứ hai là phân loại hình ảnh, mỗi hình ảnh được gắn nhãn với một trong *1000* loại, được gọi là phân loại hình ảnh. VGG 16 được đề xuất bởi Karen Simonyan và Andrew Zisserman của Phòng thí nghiệm Nhóm Hình học Trực quan của Đại học Oxford vào năm 2014 trong bài báo “MẠNG KẾT NỐI RẤT SÂU ĐỂ NHẬN DIỆN HÌNH ẢNH QUY MÔ LỚN”. Mô hình này đã giành vị trí thứ nhất và thứ hai trong các hạng mục trên trong thử thách ILSVRC 2014

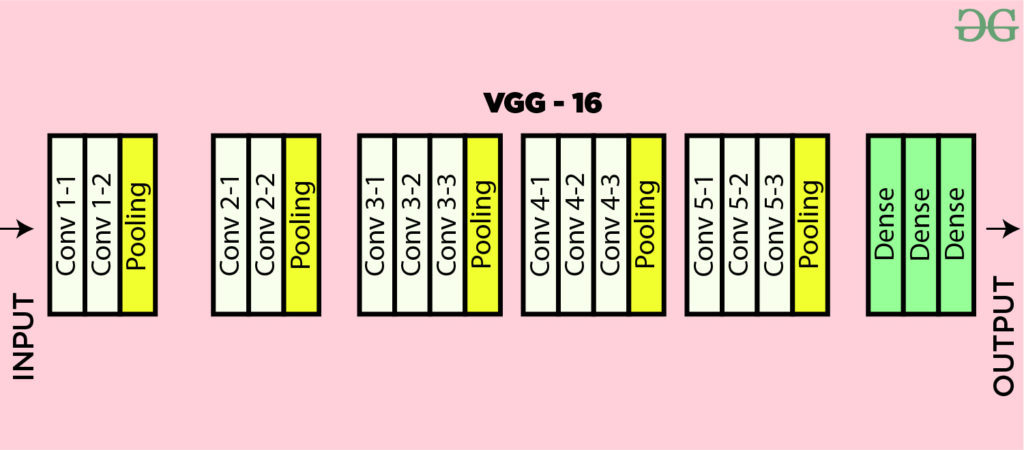
[](https://media.geeksforgeeks.org/wp-content/uploads/20200219152207/new41.jpg)

Hình 13: Kiến trúc VGG-16

Mô hình này đạt được *92,7%* độ chính xác của bài kiểm tra top 5 trên bộ dữ liệu ImageNet chứa *14* triệu hình ảnh thuộc 1000 lớp.

### **Kiến trúc VGG :**

 Đầu vào của mạng là hình ảnh có kích thước *(224, 224, 3)* . Hai lớp đầu tiên có *64* kênh với kích thước bộ lọc *3\*3* và cùng một phần đệm. Sau đó, sau một lớp nhóm tối đa có độ dài stride *(2, 2)* , hai lớp có các lớp tích chập có kích thước bộ lọc 128 và kích thước bộ lọc *(3, 3)* . Tiếp theo là lớp stride tổng hợp tối đa *(2, 2)* giống như lớp trước đó. Sau đó, có *2* lớp tích chập kích thước bộ lọc *(3, 3)* và *256* bộ lọc. Sau đó là *2* bộ *3* các lớp tích chập và một lớp nhóm tối đa. Mỗi bộ lọc có *512* bộ lọc có kích thước *(3, 3)* với cùng một phần đệm. Hình ảnh này sau đó được chuyển đến ngăn xếp của hai lớp tích chập. Trong các lớp tích chập và tổng hợp tối đa này, các bộ lọc chúng tôi sử dụng có kích thước *3\*3* thay vì *11\*11* trong AlexNet và *7\*7* trong ZF-Net. Trong một số lớp, nó cũng sử dụng pixel *1\*1 được sử dụng để thao tác số lượng kênh đầu vào.*Có một phần đệm *1 pixel* (cùng một phần đệm) được thực hiện sau mỗi lớp tích chập để ngăn tính năng không gian của hình ảnh.

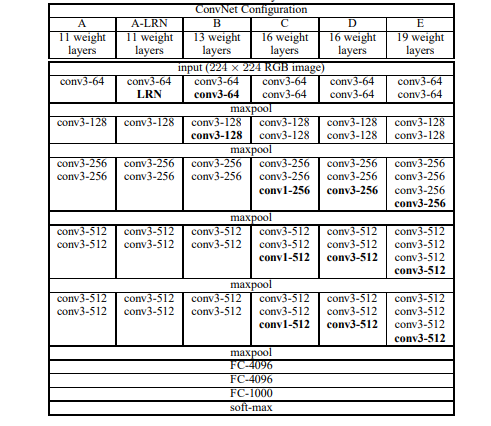
[](https://media.geeksforgeeks.org/wp-content/uploads/20200219152327/conv-layers-vgg16.jpg)

Hình 14: Bản đồ kiến trúc VGG-16

Sau khi chồng lớp tích chập và lớp tổng hợp tối đa, chúng tôi đã nhận được bản đồ tính năng *(7, 7, 512)* . Chúng tôi làm phẳng đầu ra này để biến nó thành một vectơ đặc trưng *(1, 25088)* . Sau đó, có *3* lớp được kết nối đầy đủ, lớp đầu tiên lấy đầu vào từ vectơ đặc trưng cuối cùng và xuất ra vectơ *(1, 4096)* , lớp thứ hai cũng xuất ra vectơ có kích thước *(1, 4096)* nhưng lớp thứ ba xuất ra *1000* kênh cho *1000*các lớp của thử thách ILSVRC, tức là lớp được kết nối đầy đủ thứ 3 được sử dụng để triển khai chức năng softmax để phân loại 1000 lớp. Tất cả các lớp ẩn đều sử dụng ReLU làm chức năng kích hoạt của nó. ReLU hiệu quả hơn về mặt tính toán vì nó dẫn đến việc học nhanh hơn và nó cũng làm giảm khả năng xảy ra các vấn đề về độ dốc biến mất.

### **Cấu hình:**

Bảng bên dưới liệt kê các kiến ​​trúc VGG khác nhau. Chúng ta có thể thấy rằng có 2 phiên bản VGG-16 (C và D). Không có nhiều khác biệt giữa chúng ngoại trừ một điểm trừ một số lớp tích chập, tích chập kích thước bộ lọc *(3, 3) được sử dụng thay vì (1, 1)*. Hai cái này chứa *134* triệu và *138* triệu tham số tương ứng.

[](https://media.geeksforgeeks.org/wp-content/uploads/20200217112031/VGG16conf.PNG)

Hình 15: Các cấu hình VGG khác nhau

### **Kết quả:**

VGG-16 là một trong những kiến ​​trúc hoạt động tốt nhất trong thử thách ILSVRC 2014. Nó là kiến ​​trúc về nhì trong nhiệm vụ phân loại với lỗi phân loại trong top 5 là *7,32%* (chỉ sau GoogLeNet với lỗi phân loại là *6,66%* ). Nó cũng là người chiến thắng trong nhiệm vụ nội địa hóa với *25,32%* lỗi nội địa hóa.

### **Hạn chế của VGG 16:**

* Quá trình đào tạo rất chậm (mô hình VGG ban đầu được đào tạo trên GPU Nvidia Titan trong 2-3 tuần).
* Kích thước của trọng lượng imageNet được đào tạo VGG-16 là *528* MB. Vì vậy, nó chiếm khá nhiều dung lượng ổ đĩa và băng thông khiến nó không hiệu quả.

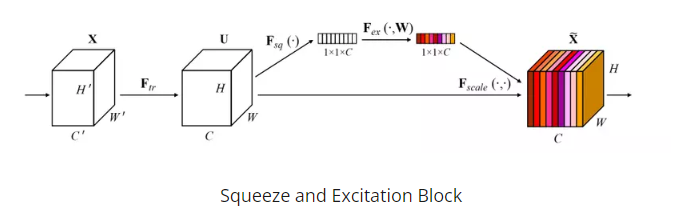
## **MobileNetV3**

Điểm cải tiến chính là việc bổ sung **Squeeze-and-Excite**.

### **Mô hình Squeeze and Excitation (SE)**

SE là một mạng khá đơn giản chỉ gồm vài lớp nhằm tăng cường thông tin giữa các kênh qua đó tăng chất lượng biểu diễn của mô hình CNN. SE làm được điều đó bằng cách sử dụng toàn bộ thông tin sau đó nhấn mạnh có chọn lọc vào từng kênh có đặc trưng quan trọng và ít chú ý vào những kênh ít quan trọng hơn. Khái niệm này khá giống với ý tưởng **Self attention** rất được hay dùng trong các bài toán xử lý ngôn ngữ tự nhiên cũng dùng đầu vào là chính nó để chú ý những thông tin quan trọng của chính nó.

### **Kiến trúc mạng**



Giải thích một số ký hiệu:

* X: ảnh đầu vào có kích thước H' x W' x C'
* *Ftr*​: tập hợp các phép biến đổi: một vài lớp convolution, hoặc 1 stage của VGG, 1 block trong ResNet, ....
* U: feature map hay đặc trưng được trính xuất từ ảnh đầu vào bởi các phép biển đổi *Ftr*​. U có kích thước H x W x C

**Bước 1:**. Ảnh đầu vào X đi qua một tập hợp các phép biển đổi *Ftr*​ trích xuất ra bản đồ đặc trưng (features map) U.

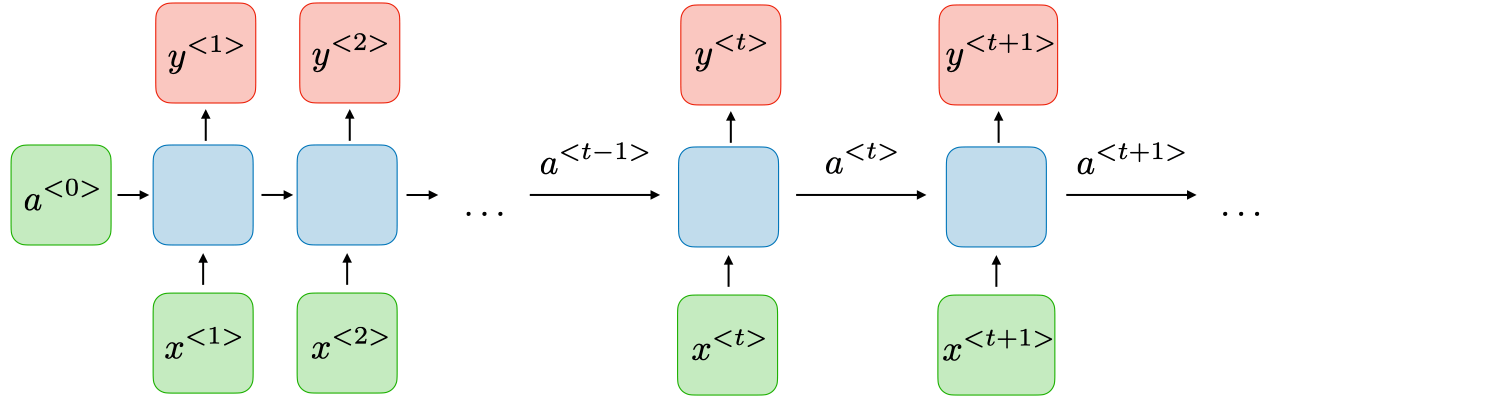
**Bước 2:**. Feature map U (H x W x C) được đi qua hàm **squeeze** sinh ra một ma trận miêu tả đặc trưng của từng kênh (1 x 1 x C) bằng cách tổng hợp features map U theo chiều H và W. Ví dụ hàm squeeze ở đây có thể là global average pooling.

**Bước 3:** Theo sau hàm squeeze là hàm excitation. Hàm excitation đóng vai trò là cơ chế miêu tả sự phụ thuộc giữa các kênh với nhau. Hàm lấy đầu vào là ma trận tổng hợp đặc trưng của từng kênh được tính toán từ bước 2 qua một vài lớp biến đổi như convolution, hàm activation, .... và cuối cùng qua hàm gate sản sinh ra trọng số chú ý cho từng kênh. Những trọng số này sau đó được nhân với feature map U để tính ra output của khối SE. Output lúc này của khối SE chỉ còn chứa những thông tin thực sự quan trọng cho bài toán. **Hàm gate ở đây thường là hàm sigmoid**.

## **Recurrent neural network (RNN)**

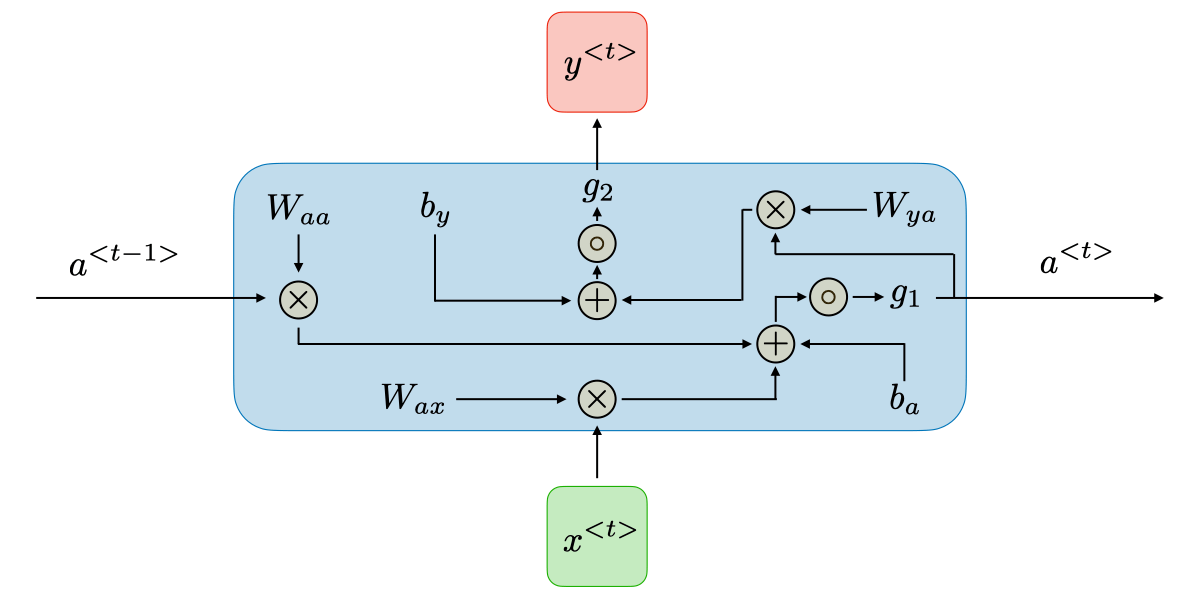
### **Kiến trúc của một mạng RNN truyền thống**

Các mạng neural hồi quy, còn được biến đến như là RNNs, là một lớp của mạng neural cho phép đầu ra được sử dụng như đầu vào trong khi có các trạng thái ẩn. Thông thường là như sau:



Tại mỗi bước *t*, giá trị kích hoạt  *a*<*t*> và đầu ra  *y*<*t*> được biểu diễn như sau:

với  ​*Waa*​,*Wya*​,*ba*​,*by*​ là các hệ số được chia sẻ tạm thời và  *g*1​,*g*2​ là các hàm kích hoạt.



Ưu và nhược điểm của một kiến trúc RNN thông thường được tổng kết ở bảng dưới đây:

|  |  |
| --- | --- |
| **Ưu điểm** | **Hạn chế** |
| • Khả năng xử lí đầu vào với bất kì độ dài nào • Kích cỡ mô hình không tăng theo kích cỡ đầu vào • Quá trình tính toán sử dụng các thông tin cũ • Trọng số được chia sẻ trong suốt thời gian | • Tính toán chậm • Khó để truy cập các thông tin từ một khoảng thời gian dài trước đây • Không thể xem xét bất kì đầu vào sau này nào cho trạng thái hiện tại |

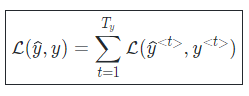
### **Ứng dụng của RNNs**

Các mô hình RNN hầu như được sử dụng trong lĩnh vực xử lí ngôn ngữ tự nhiên và ghi nhận tiếng nói. Các ứng dụng khác được tổng kết trong bảng dưới đây:

|  |  |  |
| --- | --- | --- |
| **Các loại RNN** | **Hình minh hoạ** | **Ví dụ** |
| Một-Một *Tx*​=*Ty*​=1 | https://stanford.edu/~shervine/teaching/cs-230/illustrations/rnn-one-to-one-ltr.png?9c8e3b04d222d178d6bee4506cc3f779 | Mạng neural truyền thống |
| Một-nhiều *Tx*​=1,*Ty*​>1 | https://stanford.edu/~shervine/teaching/cs-230/illustrations/rnn-one-to-many-ltr.png?d246c2f0d1e0f43a21a8bd95f579cb3b | Sinh nhạc |
| Nhiều-một *Tx*​>1,*Ty*​=1 | https://stanford.edu/~shervine/teaching/cs-230/illustrations/rnn-many-to-one-ltr.png?c8a442b3ea9f4cb81f929c089b910c9d | Phân loại ý kiến |
| Nhiều-nhiều *Tx*​=*Ty*​ | https://stanford.edu/~shervine/teaching/cs-230/illustrations/rnn-many-to-many-same-ltr.png?2790431b32050b34b80011afead1f232 | Ghi nhận thực thể tên |
| Nhiều-nhiều *Tx* ​ # *Ty*​ | https://stanford.edu/~shervine/teaching/cs-230/illustrations/rnn-many-to-many-different-ltr.png?8ca8bafd1eeac4e8c961d9293858407b | Dịch máy |

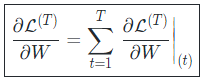
### **Hàm mất mát**

Trong trường hợp của mạng neural hồi quy, hàm mất mát L của tất cả các bước thời gian được định nghĩa dựa theo mất mát ở mọi thời điểm như sau:



### **Lan truyền ngược theo thời gian**

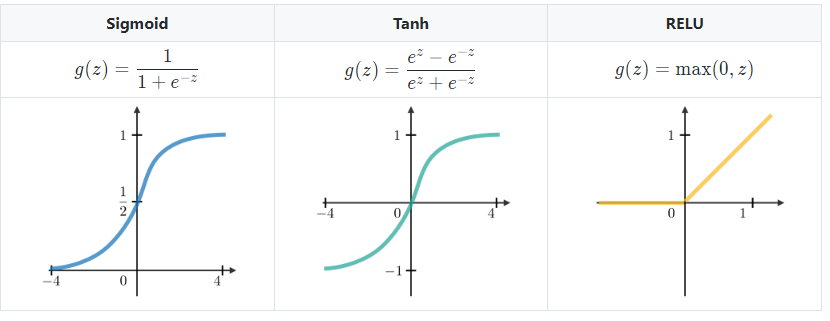
Lan truyền ngược được hoàn thành ở mỗi một thời điểm cụ thể. Ở bước *T*, đạo hàm của hàm mất mát L với ma trận trọng số *W* được biểu diễn như sau:



Xử lí phụ thuộc dài hạn

### **Các hàm kích hoạt thường dùng**

Các hàm kích hoạt thường dùng trong các modules RNN được miêu tả như sau:

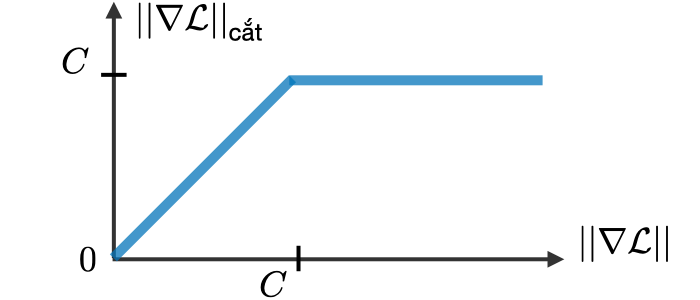


### **Vanishing/exploding gradient**

Hiện tượng vanishing và exploding gradient thường gặp trong ngữ cảnh của RNNs. Lí do tại sao chúng thường xảy ra đó là khó để có được sự phụ thuộc dài hạn vì multiplicative gradient có thể tăng/giảm theo hàm mũ tương ứng với số lượng các tầng.

### **Gradient clipping**

Là một kĩ thuật được sử dụng để giải quyết vấn đề exploding gradient xảy ra khi thực hiện lan truyền ngược. Bằng việc giới hạn giá trị lớn nhất cho gradient, hiện tượng này sẽ được kiểm soát trong thực tế.



### **Các loại cổng**

Để giải quyết vấn đề vanishing gradient, các cổng cụ thể được sử dụng trong một vài loại RNNs và thường có mục đích rõ ràng. Chúng thường được kí hiệu là Γ và bằng với:



Với W, U, b*W*,*U*,*b* là các hệ số của một cổng và \sigma*σ* là hàm sigmoid. Các loại chính được tổng kết ở bảng dưới đây:

|  |  |  |
| --- | --- | --- |
| **Loại cổng** | **Vai trò** | **Được sử dụng trong** |
| Cổng cập nhật Γ*u*​ | Dữ liệu cũ nên có tầm quan trọng như thế nào ở hiện tại? | GRU, LSTM |
| Cổng relevanc Γ*r*​ | Bỏ qua thông tin phía trước? | GRU, LSTM |
| Cổng quên Γ*f*​ | Xoá ô hay không xoá? | LSTM |
| Cổng ra Γ*o*​ | Biểu thị một ô ở mức độ bao nhiêu? | LSTM |

### **GRU/LSTM**

Gated Recurrent Unit (GRU) và Các đơn vị bộ nhớ dài-ngắn hạn (LSTM) đối phó với vấn đề vanishing gradient khi gặp phải bằng mạng RNNs truyền thống, với LSTM là sự tổng quát của GRU. Phía dưới là bảng tổng kết các phương trình đặc trưng của mỗi kiến trúc:

|  |  |  |
| --- | --- | --- |
| **Đặc tính** | **Gated Recurrent Unit (GRU)** | **Bộ nhớ dài-ngắn hạn (LSTM)** |
| *c*~<*t*> | tanh(*Wc*​[Γ*r*​⋆*a*<*t*−1>,*x*<*t*>]+*bc*​) | tanh(*Wc*​[Γ*r*​⋆*a*<*t*−1>,*x*<*t*>]+*bc*​) |
| *c*<*t*> | Γ*u*​⋆*c*~<*t*>+(1−Γ*u*​)⋆*c*<*t*−1> | Γ*u*​⋆*c*~<*t*>+Γ*f*​⋆*c*<*t*−1> |
| *a*<*t*> | *c*<*t*> | Γ*o*​⋆*c*<*t*> |
| Các phụ thuộc | https://stanford.edu/~shervine/teaching/cs-230/illustrations/gru-ltr.png?00f278f71b4833d32a87ed53d86f251c | https://stanford.edu/~shervine/teaching/cs-230/illustrations/lstm-ltr.png?4539fbbcbd9fabfd365936131c13476c |

*Chú ý: kí hiệu ⋆ chỉ phép nhân từng phần tử với nhau giữa hai vectors.*

### **Giải thích và các kí hiệu**

Có hai cách chính để biểu diễn từ được tổng kết ở bảng bên dưới:

|  |  |
| --- | --- |
| **Biểu diễn 1-hot** | **Word embedding** |
| https://stanford.edu/~shervine/teaching/cs-230/illustrations/one-hot-representation-vi.png?692834468372b64fa6f85ef5a6eb2681 | https://stanford.edu/~shervine/teaching/cs-230/illustrations/word-embedding-representation-vi.png?29fbaa278d45a46f4f31a7f36dbb5631 |
| • Lưu ý *ow*​ • Tiếp cận Naive, không có thông tin chung | • Lưu ý *ew*​ • Xem xét độ tương đồng của các từ |

### **Embedding matrix**

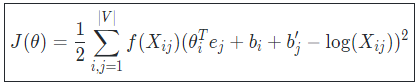
Cho một từ *w*, embedding matrix *E* là một ma trận tham chiếu thể hiện 1-hot *ow*​ của nó với embedding *ew*​ của nó như sau:



*Chú ý: học embedding matrix có thể hoàn thành bằng cách sử dụng các mô hình target/context likelihood.*

### **Word embeddings GloVe**

Mô hình GloVe, viết tắt của global vectors for word representation, nó là một kĩ thuật word embedding sử dụng ma trận đồng xuất hiện *X* với mỗi *Xi*,*j*​ là số lần mà từ đích (target) *i* xuất hiện tại ngữ cảnh *j*. Cost function *J* của nó như sau:



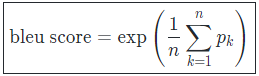
Với tính đối xứng mà *e* và *θ* có được trong mô hình này, word embedding cuối cùng *ew*(final)​ được định nghĩa như sau:



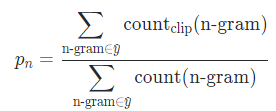
*Chú ý: Các phần tử riêng của các word embedding học được không nhất thiết là phải thông dịch được.*

### **Điểm Bleu**

Bilingual evaluation understudy (bleu) score định lượng mức độ tốt của dịch máy bằng cách tính một độ tương đồng dựa trên dự đoán n*n*-gram. Nó được định nghĩa như sau:



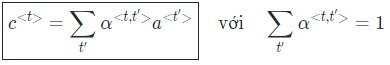
với *pn*​ là bleu score chỉ trên *n*-gram được định nghĩa như sau:



*Chú ý: một mức phạt ngắn có thể được áp dụng với các dự đoán dịch ngắn để tránh việc làm thổi phồng giá trị bleu score.*

### **Attention model**

Mô hình này cho phép một RNN tập trung vào các phần cụ thể của đầu vào được xem xét là quan trọng, nó giúp cải thiện hiệu năng của mô hình kết quả trong thực tế. Bằng việc kí hiệu *α*<*t*,*t*′> là mức độ chú ý mà đầu ra *y*<*t*> nên có đối với hàm kích hoạt *a*<*t*′> và *c*<*t*> là ngữ cảnh ở thời điểm *t*, chúng ta có:



*Chú ý: Các attention scores thường được sử dụng trong chú thích ảnh và dịch máy.*



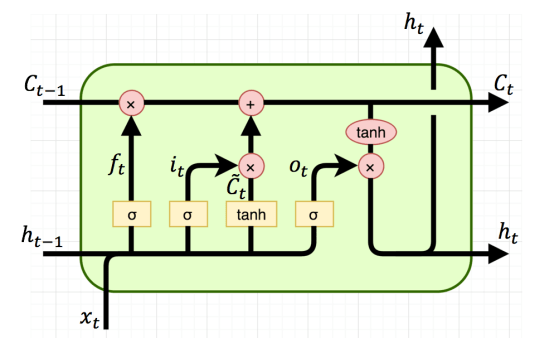


## **Long short term memory (LSTM)**

### **Mô hình LSTM**

Ở state thứ t của mô hình LSTM:

* Output: *ct*​,*ht*​, ta gọi c là cell state, h là hidden state.
* Input: *ct*−1​,*ht*−1​,*xt*​. Trong đó *xt*​ là input ở state thứ t của model. *ct*−1​,*ht*−1​ là output của layer trước. **h** đóng vai trò khá giống như **s**ở RNN, trong khi **c**là điểm mới của LSTM.



Hình 16: Mô hình LSTM

Kí hiệu *σ*, tanh ý là bước đấy dùng sigma, tanh activation function. Phép nhân ở đây là element-wise multiplication, phép cộng là cộng ma trận.

*ft*​, *it*​, *ot*​ tương ứng với **f**orget gate, **i**nput gate và **o**utput gate.

* Forget gate:  *ft*​=*σ*(*Uf*​∗*xt*​+*Wf*​∗*ht*−1​+*bf*​)
* Input gate:  *it*​=*σ*(*Ui*​∗*xt*​+*Wi*​∗*ht*−1​+*bi*​)
* Output gate: *ot*​=*σ*(*Uo*​∗*xt*​+*Wo*​∗*ht*−1​+*bo*​)

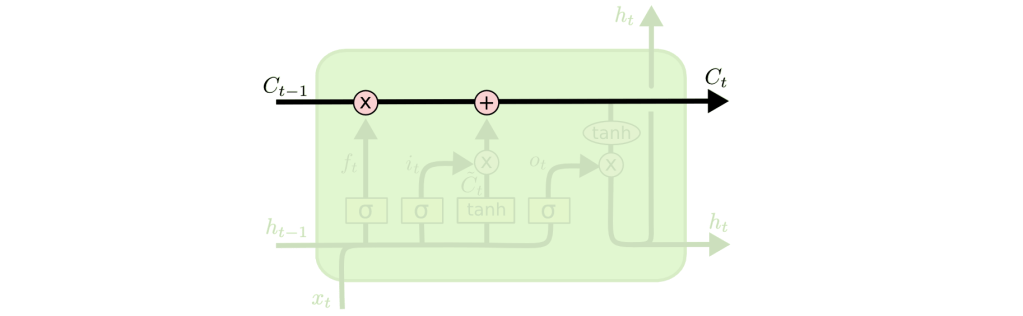
Nhận xét: 0 < *ft*​, *it*​, *ot*​<*bf*​,*bi*​,*bo*​ là các hệ số bias; hệ số W, U giống như trong bài RNN.

*ct*​~​=tanh(*Uc*​∗*xt*​+*Wc*​∗*ht*−1​+*bc*​), bước này giống hệt như tính *st*​ trong RNN.

*ct*​=*ft*​∗*ct*−1​+*it*​∗*ct*​~​, **forget gate** quyết định xem cần lấy bao nhiêu từ cell state trước và **input gate** sẽ quyết định lấy bao nhiêu từ input của state và hidden layer của layer trước.

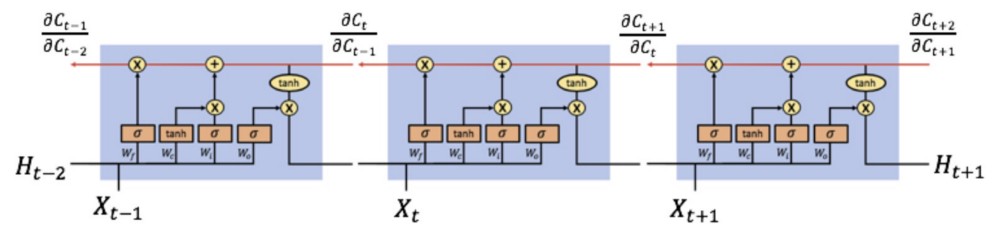
*ht*​=*ot*​∗*tanh*(*ct*​), **output gate** quyết định xem cần lấy bao nhiêu từ cell state để trở thành output của hidden state. Ngoài ra *ht*​ cũng được dùng để tính ra output *yt*​ cho state t.

**Nhận xét**: *ht*​, *ct*​~​ khá giống với RNN, nên model có **short term memory**. Trong khi đó *ct*​ giống như một băng chuyền ở trên mô hình RNN vậy, thông tin nào cần quan trọng và dùng ở sau sẽ được gửi vào và dùng khi cần => có thể mang thông tin từ đi xa=> **long term memory**. Do đó mô hình LSTM có cả short term memory và long term memory.



Hình 17: cell state trong LSTM

### **LSTM chống vanishing gradient**



Ta cũng áp dụng thuật toán back propagation through time cho LSTM tương tự như RNN.

Thành phần chính gây là vanishing gradient trong RNN là



trong đó *st*​,*W*<1.

Tương tự trong LSTM ta quan tâm đến



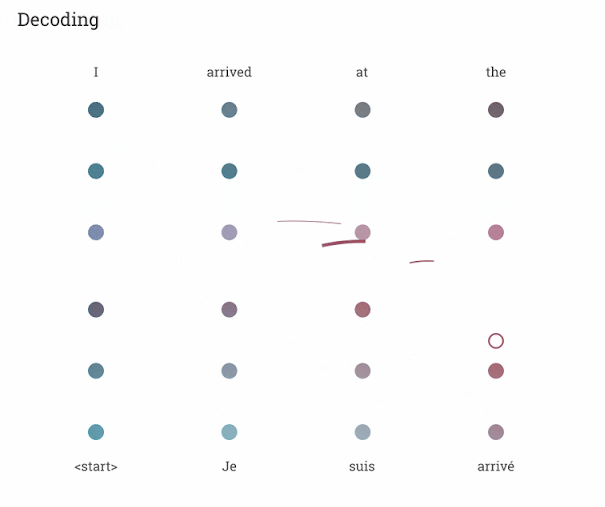
Do 0<*ft*​<1 nên về cơ bản thì LSTM vẫn bị vanishing gradient nhưng bị ít hơn so với RNN. Hơn thế nữa, khi mang thông tin trên cell state thì ít khi cần phải quên giá trị cell cũ, nên *ft*​≈1 => **Tránh được vanishing gradient**.

## **Transformers**

Transformers là mạng lưới thần kinh sâu thay thế CNN và RNN bằng tính năng tự chú ý . Tự chú ý cho phép Transformers dễ dàng truyền thông tin qua các chuỗi đầu vào.

Như đã giải thích trong bài đăng trên Google AI Blog :

Các mạng nơ-ron để dịch máy thường chứa một bộ mã hóa đọc câu đầu vào và tạo ra một biểu diễn của nó. Sau đó, bộ giải mã tạo từng câu đầu ra trong khi tham khảo biểu diễn do bộ mã hóa tạo ra. Transformer bắt đầu bằng cách tạo các biểu diễn ban đầu hoặc các phần nhúng cho mỗi từ... Sau đó, sử dụng tính năng tự chú ý, nó tổng hợp thông tin từ tất cả các từ khác, tạo ra một biểu diễn mới cho mỗi từ được thông báo bởi toàn bộ ngữ cảnh, được biểu thị bằng những quả bóng. Bước này sau đó được lặp lại nhiều lần song song cho tất cả các từ, lần lượt tạo ra các biểu diễn mới.



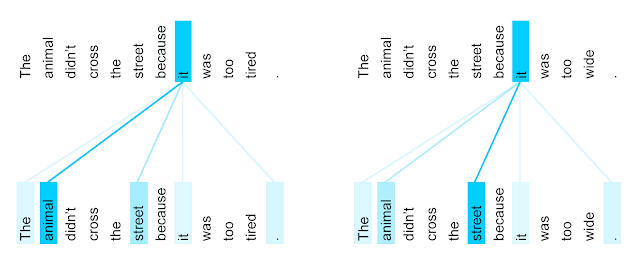
Hình 18: Ứng dụng Transformer vào dịch máy. Nguồn: Google AI Blog

Transformers là mô hình bộ mã hóa-giải mã trình tự theo trình tự tương tự như mô hình trong NMT với hướng dẫn chú ý . Máy biến áp một lớp cần thêm một chút mã để viết, nhưng gần giống với mô hình RNN của bộ mã hóa-giải mã đó. Sự khác biệt duy nhất là các lớp RNN được thay thế bằng các lớp tự chú ý. Hướng dẫn này xây dựng một Máy biến áp 4 lớp lớn hơn và mạnh hơn, nhưng về cơ bản không phức tạp hơn.

|  |  |
| --- | --- |
| **Mô hình RNN+Attention** | **Transformers 1 lớp** |
| https://www.tensorflow.org/images/tutorials/transformer/RNN+attention-words.png | https://www.tensorflow.org/images/tutorials/transformer/Transformer-1layer-words.png |

### **Tại sao Transformers lại quan trọng**

* Transformers vượt trội trong việc mô hình hóa dữ liệu tuần tự, chẳng hạn như ngôn ngữ tự nhiên.
* Không giống như các mạng thần kinh hồi quy (RNN) , Transformers có thể song song hóa được. Điều này giúp chúng hoạt động hiệu quả trên phần cứng như GPU và TPU. Lý do chính là Transformers thay thế sự lặp lại bằng sự chú ý và việc tính toán có thể xảy ra đồng thời. Đầu ra của lớp có thể được tính toán song song, thay vì theo chuỗi như RNN.
* Không giống như RNN (như seq2seq, 2014 ) hoặc mạng thần kinh tích chập (CNN) (ví dụ: ByteNet ), Transformers có thể nắm bắt các bối cảnh ở xa hoặc tầm xa và các phần phụ thuộc trong dữ liệu giữa các vị trí ở xa trong chuỗi đầu vào hoặc đầu ra. Do đó, các kết nối dài hơn có thể được học. Sự chú ý cho phép mỗi vị trí có quyền truy cập vào toàn bộ đầu vào ở mỗi lớp, trong khi ở RNN và CNN, thông tin cần phải trải qua nhiều bước xử lý để di chuyển một khoảng cách xa, điều này khiến việc học trở nên khó khăn hơn.



Hình 19: Phân phối tự chú ý của bộ mã hóa cho từ “nó” từ lớp thứ 5 đến lớp thứ 6 của Máy biến áp được đào tạo về bản dịch tiếng Anh sang tiếng Pháp (một trong tám đầu chú ý). Nguồn: Google AI Blog .

# **CHƯƠNG 3: KẾT QUẢ VÀ THỰC NGHIỆM**



## **Chuẩn bị dữ liệu cho huấn luyện**

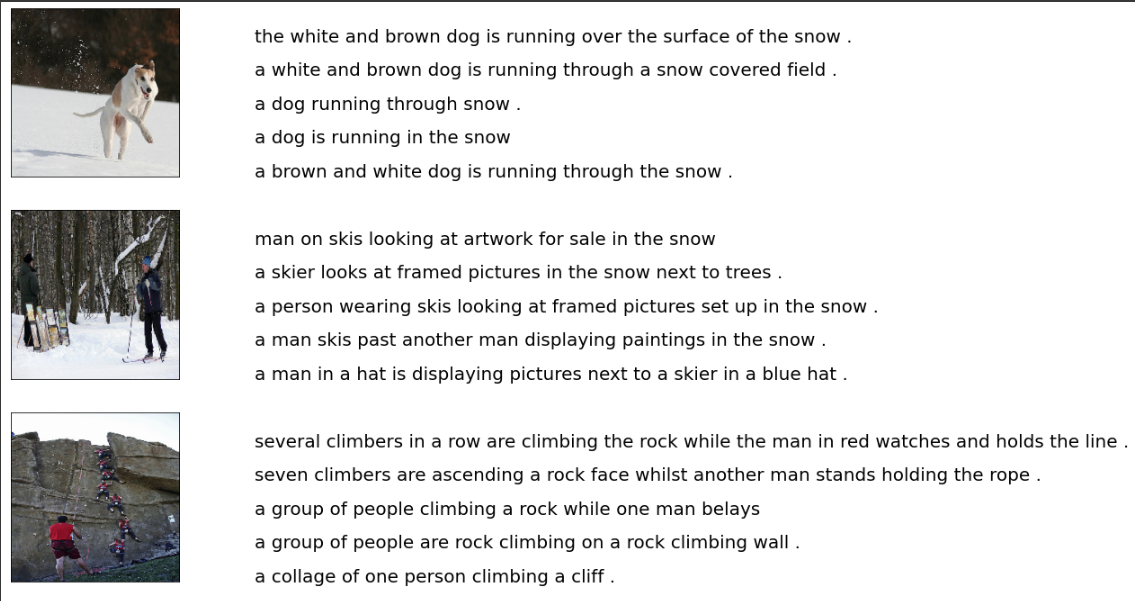
Thực hiện các thí nghiệm dựa trên tập dữ liệu đó là: Flickr8k

Các tập dữ liệu gồm 8000 các hình ảnh với mỗi ảnh có năm câu mô tả bằng tiếng Việt, tiếng Anh được mô tả bởi con người.

Chia tập dữ liệu làm hai tập gồm: Tập training, tập testing. Được chia ngẫu nhiên cho 75% cho tập trainning và 25% cho tập testing.

Tập dữ liệu training dùng để huấn luyện mô hình. Và tập testing dùng để đánh giá mô hình sau khi huấn luyện.

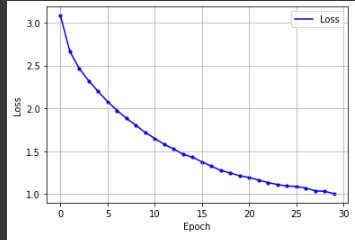
Ở bước tiền xử lý dữ liệu, chúng tôi thực hiện xây dựng bộ từ vựng, tiền xử lý câu  
mô tả và tiền xử lý ảnh. Chúng tôi xây dựng bộ từ vựng theo cách thông thường. Cụ thể, chúng tôi lấy tất cả câu mô tả của tập dữ liệu. Sau đó, chúng tôi đếm số lần xuất hiện của từng từ. Cùng với các kí tự đặc biệt <START>, <END>, <UNK>  
thì bộ từ vựng chúng tôi xây dựng cho tập dữ liệu có kích thước là 8000. Ở  
bước tiền xử lý câu mô tả, chúng tôi đưa các câu về chữ thường và làm sạch. Ở  
bước tiền xử lý ảnh, đầu tiên ta chuẩn hóa ảnh bằng cách trừ mỗi điểm ảnh cho giá trị trung bình tương ứng điểm ảnh đó; các giá trị trung bình này được thống kê trên tập dữ liệu huấn luyện (ImageNet). Tiếp theo, các ảnh có kích thước ban đầu được giảm kích thước xuống (256×256) và được cắt chính giữa với kích thước (224×224)



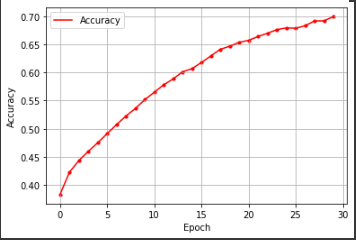
Hình 20: Tập dữ liệu và mô tả ảnh

## **Kết quả của các mô hình đã sử dụng**

### **VGG16-LSTM**

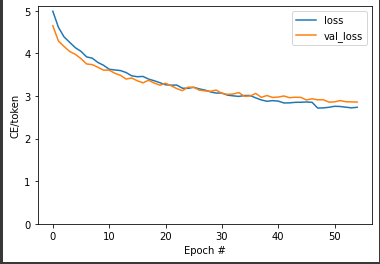


Hình 21: Đồ thị hàm Loss trên mô hình VGG16-LSTM

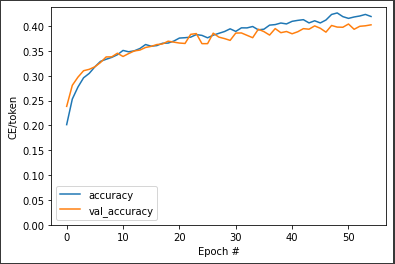


Hình 22: Đồ thị độ chính xác trên mô hình VGG16-LSTM

### **MobileNetV3 – Transformer (Attention)**

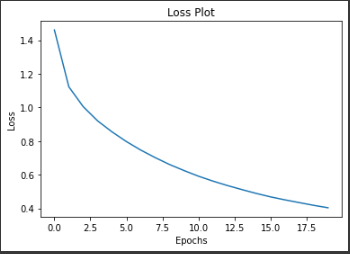


Hình 23: Đồ thị hàm Loss trên mô hình MobileNetV3-Transformer



Hình 24: Đồ thị độ chính xác trên mô hình MobileNetV3 – Transformer

### **VGG16- GRU (Attention)**



Hình 25: Đồ thị hàm Loss trên mô hình VGG16-GRU



## **So sánh độ chính xác của các mô hình.**

**Bảng :** So sánh độ chính xác giữa các mô hình

|  |  |  |  |
| --- | --- | --- | --- |
| **Mô hình** | **Độ chính xác** | **Loss** | **Epochs** |
| **VGG16-LSTM** | 0.6991 | 1.0044 | 30 |
| **MobileNetV3 – Transformer (Attention)** | 0.4024 | 2.8560 | 55 |
| **VGG16-** **GRU (Attention)** | 0.7259 | 0.4032 | 20 |

## 

## **Dự đoán mô hình**

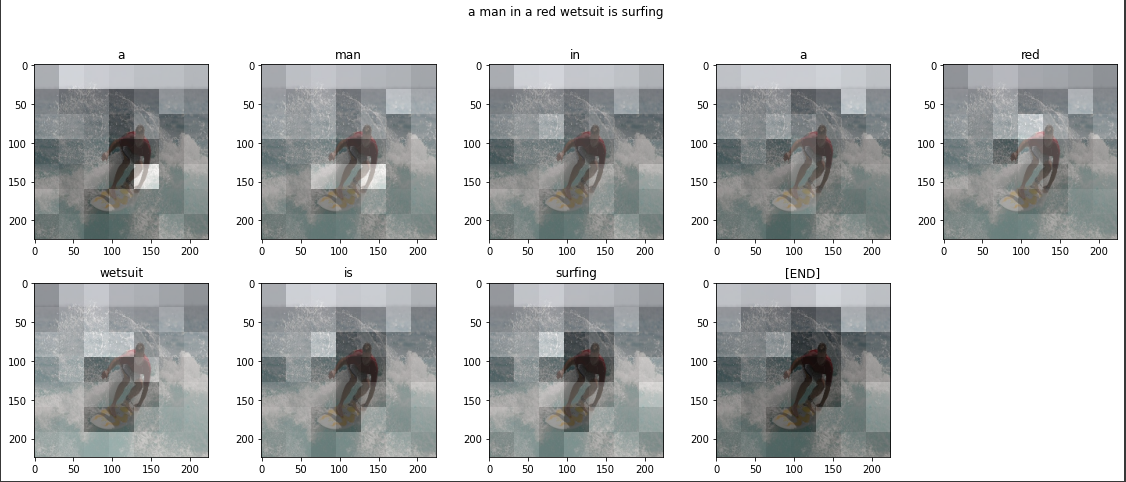
### **VGG16-LSTM**



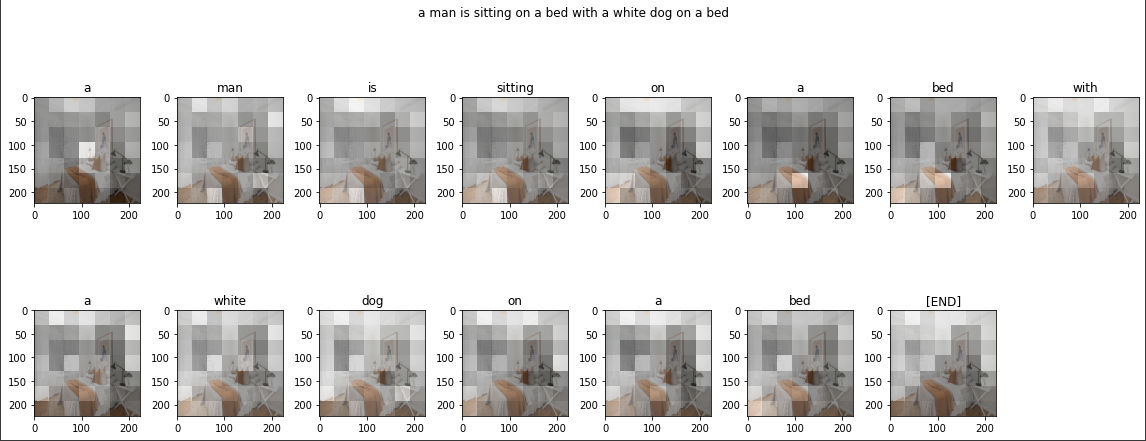
Hình 26: Kết quả dự đoán mô hình VGG16-LSTM



### MobileNetV3 – Transformer (Attention)



Hình 27: Kết quả dự đoán mô hình MobileNetV3 – Transformer (Attention)

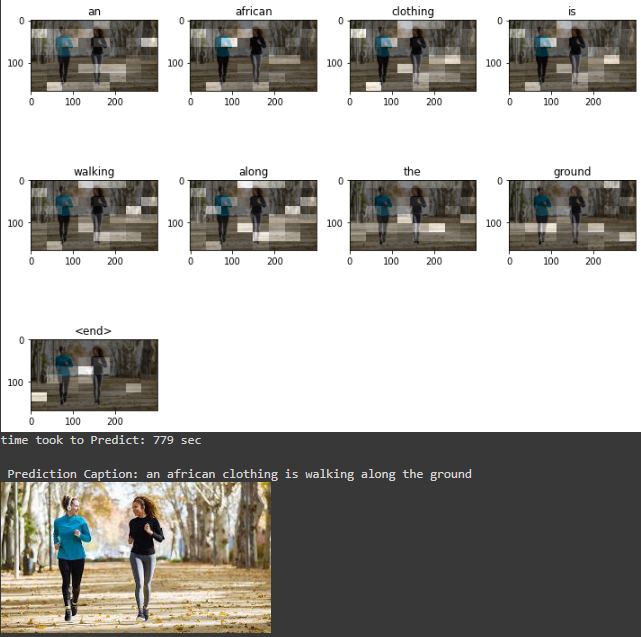


Hình 28: Kết quả dự đoán mô hình MobileNetV3 – Transformer (Attention)

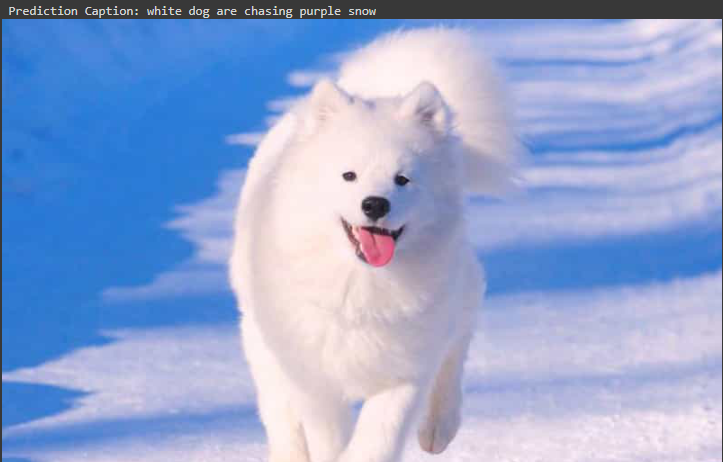
### **VGG16-GRU (Attention)**



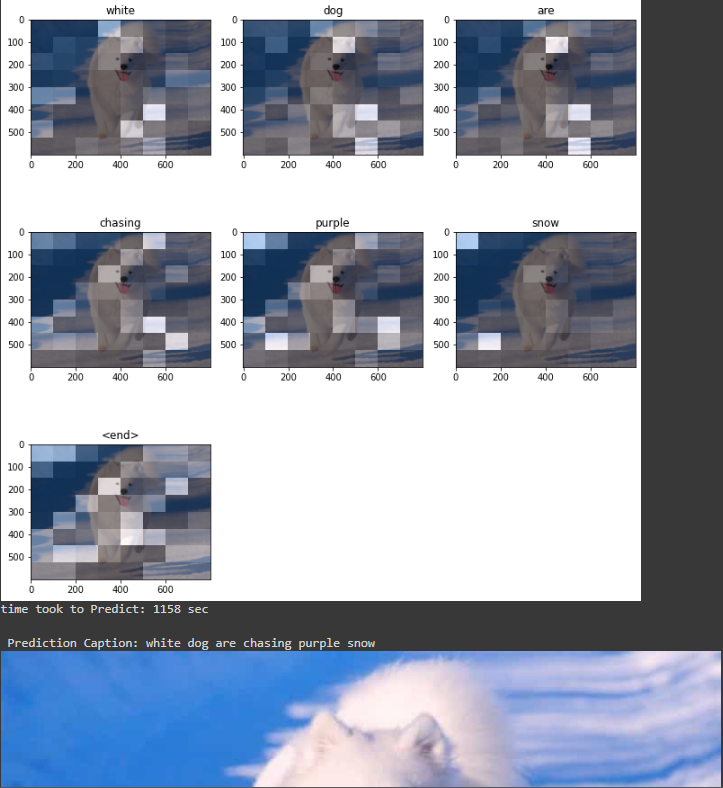
Hình 29: Kết quả dự đoán mô hình VGG16-GRU



Hình 30: Kết quả dự đoán mô hình VGG16-GRU (Attention)



Hình 31: Kết quả dự đoán mô hình VGG16-GRU



Hình 32: Kết quả dự đoán mô hình VGG16-GRU (Attention)

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



## **Kết luận**

Dựa trên các phương pháp đã có trong bài tóan phát sinh câu mô tả cho hình ảnh, nhóm đã xây dựng mô hình có hiệu quả, đạt được độ chính xác khá cao dựa trên các mô hình attention. Mô hình tập trung vào giải quyết vấn đề độ chi tiết trong các câu mô tả. Nhóm đã chạy thử nghiệm và đánh giá kết quả trên tập dữ liệu Flick8k với các độ đo đánh giá khác nhau. Kết quả cho thấy mô hình đạt được sự cải tiến nhất định khi kết quả mô tả tốt hơn một số mô hình hiện có và mô hình có thể phát sinh ra những câu mô tả với độ chính xác cao. Việc phát sinh mô tả ảnh tự động giúp giảm bớt gánh nặng gán nhãn thủ công và hỗ trợ tốt hơn cho người khiếm thị

## **Hạn chế**

* Việc phân loại và dự đoán vẫn còn chưa chính xác hoàn toàn trên nhiều hình ảnh.
* Tập dữ liệu vẫn chưa được phong phú về số lượng ảnh.
* Hệ thống chỉ dừng lại ở mức nghiên cứu, chưa thể đưa ra áp dụng thực tế và chưa đủ chức năng để tới tay người dùng

## **Hướng phát triển**

* Cải tiến dữ liệu
* Cải tiến phương pháp, mô hình cho tăng độ chính xác cao hơn
* Nâng cấp và hoàn thiện chương trình
* Cải thiện tốc độ xử lý trên hệ thống

# TÀI LIỆU THAM KHẢO

[1] Giới thiệu về mạng CNN

<https://towardsdatascience.com/a-comprehensive-guide-to-convolutional-neural-networks-the-eli5-way-3bd2b1164a53>

[2] Model LSTM

<https://nttuan8.com/bai-14-long-short-term-memory-lstm/>

[3] Mạng RNN

<https://stanford.edu/~shervine/l/vi/teaching/cs-230/cheatsheet-recurrent-neural-networks>

[4] Model VGG16

<https://www.geeksforgeeks.org/vgg-16-cnn-model/>

[5] Y. LeCun, Y. Bengio, and G. Hinton, “Deep learning,” in Nature, 2015. 10

[6] Flick8k Dataset

<https://www.kaggle.com/datasets/adityajn105/flickr8k>

[7] Keiron O’Shea 1 and Ryan Nash 2 (2015), “Introduction to Convolutional Neural Networks”

<https://arxiv.org/pdf/1511.08458.pdf>

[8] Show, Attend and Tell: Neural Image Caption Generation with Visual Attention

<https://arxiv.org/pdf/1502.03044v3.pdf>

[9] I. G. Y. Bengio and A. Courville, “Deep learning,” 2016, book in preparation  
for MIT Press. [Online]. Available: <https://www.deeplearningbook.org>

[10] O. Vinyals, A. Toshev, S. Bengio, and D. Erhan, “Show and tell: A neural image caption generator,” in Proceedings of the IEEE conference on computer visionand pattern recognition, pp. 3156–3164, 2015

[11] Model MobileNetV3

<https://viblo.asia/p/cnn-architecture-series-1-mobilenets-mo-hinh-gon-nhe-cho-mobile-applications-1VgZvJV1ZAw>

[12] Model Transformer

<https://www.tensorflow.org/text/tutorials/transformer#why_transformers_are_significant>