BỘ GIÁO DỤC VÀ ĐÀO TẠO BỘ NÔNG NGHIỆP VÀ MÔI TRƯỜNG
TRƯỜNG ĐẠI HỌC THỦY LỢI

ĐỚI XUÂN ĐẠT

XÂY DỰNG MÔ HÌNH HỌC SÂU ĐA PHƯƠNG THỨC CHO NHIỆM VỤ HỎI ĐÁP DỰA TRÊN HÌNH ẢNH

ĐỒ ÁN TỐT NGHIỆP

HÀ NỘI, NĂM 2025

BỘ GIÁO DỤC VÀ ĐÀO TẠO BỘ NÔNG NGHIỆP VÀ MÔI TRƯỜNG
TRƯỜNG ĐẠI HỌC THỦY LỢI

ĐỚI XUÂN ĐẠT

XÂY DỰNG MÔ HÌNH HỌC SÂU ĐA PHƯƠNG THỨC CHO NHIỆM VỤ HỎI ĐÁP DỰA TRÊN HÌNH ẢNH

Ngành: Công nghệ thông tin
Mã số: 7480103

NGƯỜI HƯỚNG DẪN 1. ThS. Trương Xuân Nam

HÀ NỘI, NĂM 2025

CỘNG HOÀ XÃ HỘI CHỦ NGHĨA VIỆT NAM
Độc lập - Tự do - Hạnh phúc
---------★---------

NHIỆM VỤ ĐỒ ÁN TỐT NGHIỆP

Họ tên sinh viên: Đới Xuân Đạt Hệ đào tạo: Chính quy

Lớp: 61THNB Ngành: Công nghệ thông tin

Khoa: Công nghệ thông tin

1- TÊN ĐỀ TÀI:

XÂY DỰNG MÔ HÌNH HỌC SÂU ĐA PHƯƠNG THỨC CHO NHIỆM VỤ HỎI ĐÁP DỰA TRÊN HÌNH ẢNH

2- CÁC TÀI LIỆU THAM KHẢO:

- Cem Akkus, Luyang Chu, Vladana Djakovic, et al, "Multimodal Deep Learning" arXiv:2301.04856 , 2023.
- Lucas Beyer, Andreas Steiner, André Susano Pinto, et al, "PaliGemma: A versatile 3B VLM for transfer", arXiv:2407.07726, 2024.
- Alec Radford, Jong Wook Kim, Chris Hallacy, et al, "Learning Transferable Visual Models From Natural Language Supervision", arXiv:2103.00020, 2021.
- Xiaohua Zhai, Basil Mustafa, Alexander Kolesnikov, et al, "Sigmoid Loss for Language Image Pre-Training", arXiv:2303.15343, 2023.
- A. Dosovitskiy, L. Beyer, A. Kolesnikov, et al, "An Image is Worth 16x16 Words: Transformers for Image Recognition at Scale", arXiv:2010.11929, 2021.
- Ashish Vaswani, Noam Shazeer, Niki Parmar, et al, "Attention Is All You Need", arXiv:1706.03762, 2017.
- Kaiming He, Xiangyu Zhang, Shaoqing Ren, et al, "Deep Residual Learning for Image Recognition", arXiv:1512.03385, 2015.
- Jimmy Lei Ba, Jamie Ryan Kiros, Geoffrey E. Hinton, "Layer Normalization", arXiv:1607.06450, 2016.
- Noam Shazeer, "Fast Transformer Decoding: One Write-Head is All You Need", arXiv:1911.02150, 2019.
- Jianlin Su, Yu Lu, Shengfeng Pan, et al, "RoFormer: Enhanced Transformer with Rotary Position Embedding", arXiv:2104.09864, 2021.
- Noam Shazeer, "GLU Variants Improve Transformer", arXiv:2002.05202, 2020.
- Dan Hendrycks, Kevin Gimpel, "Gaussian Error Linear Units (GELUs)", arXiv:1606.08415, 2023.

- Biao Zhang, Rico Sennrich, "Root Mean Square Layer Normalization", arXiv:1910.07467, 2019.
- Aishwarya Agrawal, Jiasen Lu, Stanislaw Antol, et al, "VQA: Visual Question Answering", arXiv:1505.00468, 2015.
- merve, Andreas P. Steiner, Pedro Cuenca, "PaliGemma – Google's Cutting-Edge Open Vision Language Model", 2024, [Online]. Available: https://huggingface.co/blog/paligemma

3- NỘI DUNG CÁC PHẦN THUYẾT MINH VÀ TÍNH TOÁN: Tỷ lệ %

- Chương 1: Giới thiệu tổng quan bài toán 10%
- Chương 2: Cơ sở lý thuyết 40%
- Chương 3: Xây dựng Mô hình hỏi đáp dựa trên hình ảnh 20%
- Chương 4: Thực nghiệm và đánh giá 20%
- Chương 5: Kết luận và phương hướng phát triển 10%

4- GIÁO VIÊN HƯỚNG DẪN TỪNG PHẦN

Giáo viên hướng dẫn toàn bộ đồ án: ThS.Trương Xuân Nam

5- NGÀY GIAO NHIỆM VỤ ĐỒ ÁN TỐT NGHIỆP

Ngày ……tháng ……năm 2025

Trưởng Bộ môn
(Ký và ghi rõ Họ tên)

Giáo viên hướng dẫn chính
(Ký và ghi rõ Họ tên)

Nhiệm vụ Đồ án tốt nghiệp đã được Hội đồng thi tốt nghiệp của Khoa thông qua

Ngày……tháng……năm 2025

Chủ tịch Hội đồng
(Ký và ghi rõ Họ tên)

Sinh viên đã hoàn thành và nộp bản Đồ án tốt nghiệp cho Hội đồng thi ngày……tháng……năm 2025

Sinh viên làm Đồ án tốt nghiệp
(Ký và ghi rõ Họ tên)

TRƯỜNG ĐẠI HỌC THUỶ LỢI
KHOA CÔNG NGHỆ THÔNG TIN

BẢN TÓM TẮT ĐỀ CƯƠNG ĐỒ ÁN TỐT NGHIỆP

TÊN ĐỀ TÀI: **XÂY DỰNG MÔ HÌNH HỌC SÂU ĐA PHƯƠNG THỨC CHO NHIỆM VỤ HỎI ĐÁP DỰA TRÊN HÌNH ẢNH**

Sinh viên thực hiện: Đới Xuân Đạt
Lớp: 61THNB
Giáo viên hướng dẫn: ThS. Trương Xuân Nam

TÓM TẮT ĐỀ TÀI

Nhiệm vụ hỏi đáp dựa trên hình ảnh (Visual Question Answering) là một bài toán quan trọng và đầy thách thức trong lĩnh vực trí tuệ nhân tạo đa phương thức. Bài toán yêu cầu xây dựng các mô hình có khả năng kết hợp thông tin từ cả dữ liệu hình ảnh lẫn ngôn ngữ tự nhiên để đưa ra câu trả lời chính xác. Nguồn cảm hứng của em cho đề tài đến từ sự phát triển mạnh mẽ của các mô hình học sâu và nhu cầu thực tế về các hệ thống thông minh có khả năng nhìn và hiểu như con người.

Bài toán đặt ra ở đây là xây dựng mô hình có thể trả lời câu hỏi về một bức ảnh. Quá trình này đòi hỏi mô hình phải thực hiện ba tác vụ chính: thứ nhất, phân tích hình ảnh để nắm bắt các đối tượng và mối quan hệ trong đó; thứ hai, hiểu câu hỏi để xác định thông tin cần truy vấn; và thứ ba, tích hợp thông tin thị giác và ngôn ngữ một cách hiệu quả để đưa ra câu trả lời cuối cùng. Khả năng thực hiện tốt chu trình này là yếu tố then chốt và là thách thức chính của bài toán em đặt ra

Hướng tiếp cận chính của đề tài là nghiên cứu và phát triển mô hình VQA dựa trên các kỹ thuật học sâu, sử dụng bộ dữ liệu chuẩn có tên Visual Question Answering (VQA) version 2.0. Quá trình thực hiện sẽ tập trung vào việc nghiên cứu, lựa chọn và huấn luyện các kiến trúc mô hình Vision Language phù hợp, trong đó có thể bao gồm việc fine-tuning các mô hình mạnh như Paligemma và điều chỉnh kiến trúc để tối ưu hóa hiệu năng trên bộ dữ liệu này. Các bước chính bao gồm tiền xử lý dữ liệu, thiết kế, cấu hình mô hình, huấn luyện và đánh giá hiệu năng.

Nhiệm vụ VQA có nhiều ứng dụng thực tế, chẳng hạn như trong các hệ thống hỗ trợ người khiếm thị, phân tích hình ảnh y tế, trợ lý ảo thông minh, và các ứng dụng giáo dục tương tác... Từ đó em quyết định chọn đề tài **"Xây dựng mô hình học sâu đa phương thức cho nhiệm vụ hỏi đáp dựa trên hình ảnh."**

Công nghệ sử dụng:

- Ngôn ngữ lập trình: Python
- Học sâu cho ngôn ngữ tự nhiên và thị giác máy tính
- Fine-tuning pretrained model

CÁC MỤC TIÊU CHÍNH

Mục tiêu 1. Tìm hiểu bài toán Hỏi đáp dựa trên hình ảnh (VQA) và các phương pháp tiếp cận

Mục tiêu 2. Nghiên cứu các kiến trúc mô hình Vision-Language và lựa chọn mô hình phù hợp với bài toán

Mục tiêu 3. Chuẩn bị và xử lý bộ dữ liệu VQA v2.0 cho việc huấn luyện mô hình.

Mục tiêu 4. Thực hiện huấn luyện mô hình VQA theo hướng tiếp cận đã chọn trên bộ dữ liệu VQA v2.0

Mục tiêu 5. Đánh giá hiệu năng của mô hình đã huấn luyện bằng phép đo VQA Accuracy

Mục tiêu 6. Kết luận và viết báo cáo tổng kết

KẾT QUẢ DỰ KIẾN

- Hoàn thành các mục tiêu đã đề ra
- Xây dựng được mô hình học máy có thể trả lời hỏi đáp dựa trên hình ảnh
- Báo cáo tổng kết Đồ Án Tốt Nghiệp

LỜI CAM ĐOAN

Em xin cam đoan đây là Đồ án tốt nghiệp của cá nhân em. Các kết quả trong Đồ án tốt nghiệp này là trung thực và thực hiện dưới sự hướng dẫn của thầy ThS.Trương Xuân Nam. Việc tuân thủ quy định về trích dẫn và ghi nguồn tài liệu tham khảo đã được thực hiện sau khi tham khảo các nguồn tài liệu (nếu có).

Tác giả ĐATN

Chữ ký

Đới Xuân Đạt

LỜI CÁM ƠN

Qua quãng thời gian học tập, rèn luyện và nghiên cứu tại Khoa Công Nghệ Thông Tin Trường Đại Học Thủy Lợi, bản thân em đã trải qua một chặng đường học với một môi trường đào tạo chất lượng, được hướng dẫn bởi những thầy cô tận tâm và yêu quý.

Em xin gửi lời cảm ơn chân thành đến quý thầy cô tại Trường Đại học Thủy Lợi, người đã trang bị cho em những kiến thức vô cùng quý báu suốt bốn năm học tập tại đây. Đặc biệt, lòng nhiệt huyết và sự chuyên nghiệp của các thầy cô trong Khoa Công nghệ thông tin đã tận tình giảng dạy, chỉ bảo và trang bị cho em những kiến thức chuyên ngành vô cùng quý giá, giúp em có đủ nền tảng để hoàn thành Đồ án tốt nghiệp này. Em chân thành biết ơn sự đồng hành và sự hỗ trợ quý báu từ các thầy cô trong suốt hành trình học tập của mình. Em muốn bày tỏ lòng biết ơn sâu sắc đặc biệt đến TS. Trương Xuân Nam, giảng viên tận tình của Trường Đại học Thủy Lợi. Từ những ngày đầu tiên xây dựng đồ án, thầy đã không ngừng hướng dẫn em, đưa ra những góp ý chi tiết và truyền đạt những kinh nghiệm quý báu. Những đóng góp này đã giúp em vượt qua những thách thức và hoàn thành Đồ án tốt nghiệp một cách xuất sắc.

Em xin gửi lời cảm ơn sâu sắc nhất đến TS. Trương Xuân Nam vì sự tận tâm và sự hỗ trợ quý báu trong quá trình nghiên cứu và thực hiện đồ án của mình. Qua quá trình hoàn thành đồ án, em nhận thức rõ ràng kiến thức của mình vẫn còn hạn chế và thiếu kinh nghiệm thực tế. Điều này là không tránh khỏi và em tự nhận ra những điểm chưa hoàn hảo. Vì lẽ đó, em chân thành kính mong nhận được những ý kiến đóng góp quý báu từ quý thầy cô để em có cơ hội hoàn thiện và nâng cao sâu rộng kiến thức của mình trong lĩnh vực này. Em xin chân thành cảm ơn vì sự quan tâm và hỗ trợ từ phía quý thầy cô!

MỤC LỤC

DANH MỤC CÁC HÌNH ẢNH

DANH MỤC TỪ VIẾT TẮT

AI Artificial Intelligence

CNN Convolutional Neural Network

LLM Large Language Model

LSTM Long Short-Term Memory

LVM Language Vision Model

NLP Natural Language Processing

ViT Vision Transformer

VQA Visual Question Answering

CHƯƠNG 1: Giới thiệu tổng quan bài toán

1.1. Lý do chọn đề tài

Trí tuệ nhân tạo đang ngày càng hướng tới việc mô phỏng khả năng hiểu thêm về thế giới của con người, trong đó việc kết hợp thông tin thị giác và ngôn ngữ là một thách thức then chốt. Bài toán VQA là một phép thử quan trọng cho năng lực này, đòi hỏi mô hình AI phải nhìn ảnh và hiểu câu hỏi để đưa ra câu trả lời chính xác. Sự phát triển nhanh chóng của các Mô hình thị giác-ngôn ngữ dựa trên kiến trúc Transformer đã mang lại những kết quả ấn tượng cho VQA. Tuy nhiên, nhiều mô hình hiệu năng cao đòi hỏi tài nguyên tính toán khổng lồ, đặt ra bài toán về hiệu quả và khả năng tiếp cận. Sau khi tiến hành khảo sát các kiến trúc hàng đầu trong lĩnh vực, PaliGemma-3B nổi lên như một lựa chọn chiến lược và phù hợp nhất với mục tiêu của bài toán. Trong khi các mô hình khác hoặc có kích thước quá lớn gây khó khăn cho việc tinh chỉnh, hoặc đã bị các kiến trúc thế hệ mới vượt qua về mặt hiệu năng, PaliGemma-3B lại chiếm một vị trí rất đặc biệt. Đề tài quyết định lựa chọn mô hình này vì các lý do chính sau:

- Hiệu năng đã được kiểm chứng: Kiến trúc PaliGemma đã chứng tỏ được hiệu năng cạnh tranh ở cấp cao nhất trên các benchmark tiêu chuẩn của bài toán VQA, đảm bảo rằng hướng nghiên cứu của đề tài đang làm việc với một phương pháp hiện đại và hiệu quả.
- Cân bằng tối ưu giữa hiệu năng và tài nguyên: Với kích thước 3 tỷ tham số, mô hình này đạt được sự cân bằng xuất sắc, cho phép thực hiện quá trình fine-tune trong điều kiện tài nguyên giới hạn mà không phải hy sinh quá nhiều về mặt hiệu năng.
- Kiến trúc phù hợp và cộng đồng hỗ trợ mạnh mẽ: Mô hình được thiết kế chuyên biệt cho các tác vụ hiểu đa phương thức ở cấp độ tiền tố (prefix-level), rất phù hợp với bản chất của bài toán VQA. Đồng thời, sự hỗ trợ từ cộng đồng Hugging Face giúp quá trình triển khai trở nên thuận lợi hơn.

Vì vậy, đề tài này được lựa chọn nhằm nghiên cứu, triển khai và đánh giá hiệu năng của mô hình PaliGemma cho bài toán VQA. Mục tiêu là tìm hiểu sâu về cách tiếp cận mới

này, phân tích khả năng của nó trên các bộ dữ liệu VQA chuẩn, qua đó đóng góp cái nhìn cập nhật về một hướng phát triển quan trọng và có tính ứng dụng cao trong lĩnh vực AI đa phương thức. Đồng thời, đây cũng là cơ hội để em đi sâu vào lĩnh vực đang rất được quan tâm này.

1.2. Thách thức của hỏi đáp dựa trên hình ảnh

Nhiệm vụ hỏi đáp dựa trên hình ảnh đòi hỏi sự kết hợp giữa thị giác máy tính và xử lý ngôn ngữ tự nhiên. Hệ thống không chỉ cần hiểu biết sâu sắc về nội dung hình ảnh, nhận diện đối tượng, mối quan hệ không gian phức tạp, mà còn phải diễn giải chính xác ý định và ngữ nghĩa của các câu hỏi ngôn ngữ tự nhiên, vốn rất đa dạng và đôi khi mơ hồ. Thách thức cốt lõi nằm ở khả năng suy luận phức hợp để liên kết thông tin thị giác và văn bản, bao gồm suy luận logic, suy luận dựa trên kiến thức phổ thông và đôi khi là kiến thức chuyên ngành, nhằm đưa ra câu trả lời tự nhiên và chính xác. Bên cạnh đó, sự đa dạng của dữ liệu hình ảnh và câu hỏi trong thực tế, cùng với yêu cầu về độ chính xác cao và khả năng giải thích, càng làm tăng độ khó của bài toán. Giải quyết hiệu quả các thách thức này đóng góp quan trọng vào việc phát triển các hệ thống AI thông minh hơn, có khả năng tương tác và hiểu thế giới đa phương tiện một cách toàn diện và hiệu quả hơn.

Hình 1.1. Các ví dụ về câu hỏi dạng mở, tự do

1.3. Mục tiêu nghiên cứu

Mục tiêu nghiên cứu chính của đồ án này là tìm hiểu sâu về các phương pháp và kỹ thuật trong việc hỏi đáp thông qua hình ảnh, từ đó đề xuất và xây dựng một mô hình đa phương thức có khả năng hiểu và suy luận hiệu quả trên dữ liệu hình ảnh. Đồ án sẽ tập trung vào việc nghiên cứu các kiến trúc học sâu của cả Thị giác máy tính (Computer Vision) và Xử lý ngôn ngữ tự nhiên (Natural Language Processing), cách thức kết hợp thông tin đa phương thức, nghiên cứu hướng đến việc đánh giá và so sánh hiệu năng của mô hình đề xuất, đồng thời cung cấp những hiểu biết thực tiễn về việc xây dựng và tối ưu hóa các hệ thống AI tương tác thông minh.

1.4. Phạm vi nghiên cứu

Phạm vi nghiên cứu của đồ án này sẽ tập trung vào việc ứng dụng và đánh giá các mô hình học sâu cho bài toán trên bộ dữ liệu VQA v2. Nghiên cứu sẽ giới hạn trong việc tìm hiểu các kiến trúc mô hình đa phương thức. Việc trích xuất đặc trưng hình ảnh và biểu diễn câu hỏi sẽ sử dụng các phương pháp và mô hình tiền huấn luyện (pre-trained models) sẵn có để tập trung vào khía cạnh suy luận và kết hợp thông tin. Đồ án sẽ thực nghiệm với việc huấn luyện, tinh chỉnh và đánh giá hiệu năng của các mô hình đã chọn theo quy chuẩn của bộ data VQA

CHƯƠNG 2: Cơ sở lý thuyết

Bài toán hỏi đáp dựa trên hình ảnh là bài toán khó của lĩnh vực AI. Ta điều biết để mô hình có khả năng suy luận, xác định được ngữ cảnh cũng như các mối quan hệ giữa các vật thể trong hình ảnh thực tế và trả lời được các câu hỏi liên quan một cách tiệm cận con người nhất có thể là một thách thức rất lớn. Chương này sẽ tập trung về vấn kỹ thuật xoay quanh đến việc giải quyết bài toán Hỏi đáp dựa trên hình ảnh, tìm hiểu các nghiên cứu trong nước cũng như của nước ngoài đã và đang thực nghiệm.

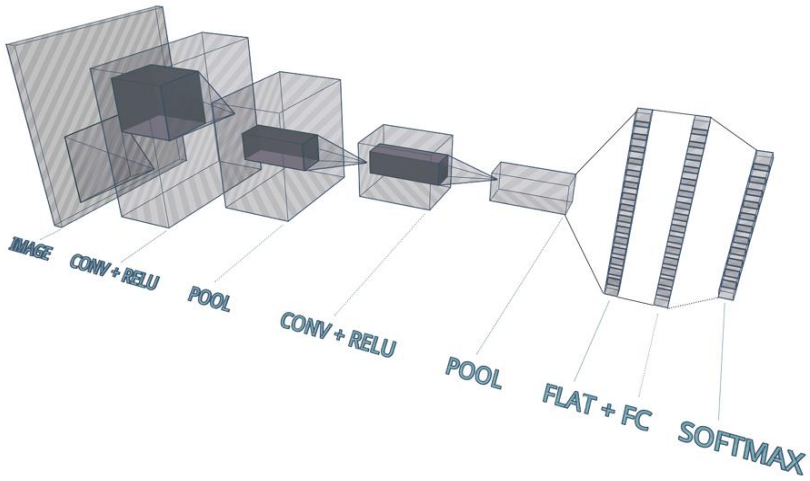
2.1. Các công nghệ nền tảng

Sự phát triển của các mô hình Hỏi đáp dựa trên hình ảnh hiện đại được xây dựng trên một vài trụ cột công nghệ nền tảng, vốn đã định hình lại toàn bộ lĩnh vực Trí tuệ Nhân tạo. Việc hiểu rõ các kiến trúc này là điều cần thiết để có thể phân tích các hệ thống VQA phức tạp. Trước khi đi sâu vào chi tiết, chúng ta sẽ đi qua về một số khái niệm học sâu như Xử lý Ngôn ngữ Tự nhiên và Thị giác Máy tính. Những khái niệm này sẽ giúp ta hiểu rõ hơn về bản chất của các hệ thống này.

2.1.1. Các phương pháp cổ điển

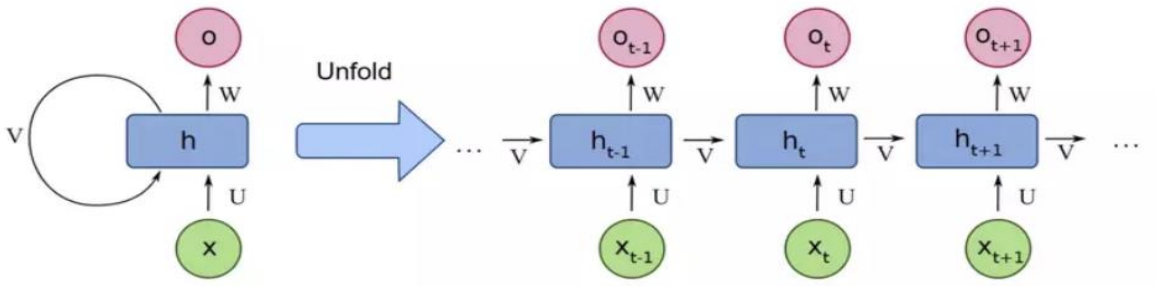
Computer Vision tập trung vào việc mô phỏng sự phức tạp của hệ thống thị giác con người, giúp máy tính có khả năng nhận diện và xử lý đối tượng trong hình ảnh, video tương tự như cách con người thực hiện. Trong những năm gần đây, Computer Vision đã trở thành một trong những lĩnh vực thiết yếu và có ứng dụng sâu rộng trong khoa học máy tính. Một trong số đó là bài toán tối ưu hóa các mạng nơ-ron tích chập sâu (deep convolutional neural networks) cho nhiệm vụ image classification. Độ chính xác của quá trình phân loại này phụ thuộc vào các yếu tố như độ rộng, độ sâu của mạng và độ phân giải của hình ảnh. Để giải quyết tình trạng suy giảm độ chính xác trong quá trình huấn luyện như overfitting, vanishing gradients, exploding gradients,.. một hướng tiếp cận đó là kiến trúc deep residual learning. Một hướng khác, tuy ít phổ biến hơn nhưng cũng hiệu quả để tăng độ chính xác, là mở rộng quy mô của ConvNets thông qua việc tăng độ phân giải hình ảnh. Dựa trên nhận định này, một phương pháp nhân rộng

kết hợp đơn giản mà hiệu quả mang tên EfficientNets do Tiến sĩ Lê Viết Quốc và Mingxing Tan đề xuất. [1]



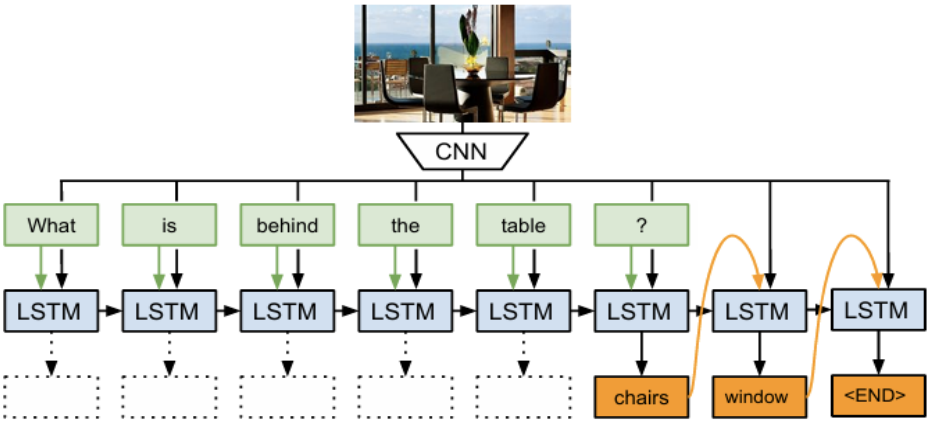
Hình 2.1: Cấu trúc chung của một mạng CNN

Xử lý Ngôn ngữ Tự nhiên, dù đã hình thành khoảng nửa thế kỷ, nhưng vai trò của nó ngày càng trở nên thiết yếu. NLP vốn tập trung vào ngôn ngữ nói và viết, đã chứng kiến nhiều bước đột phá. Chẳng hạn, việc học các biểu diễn nội bộ của từ (internal representations of words) là một trong những thành tựu nổi bật của thập kỷ vừa qua. Chính Word embeddings đã hiện thực hóa điều này, nó cho phép các developer mã hóa từ ngữ thành những vector dày đặc (dense vectors) cô đọng nội dung ngữ nghĩa bên trong. Nhờ đó, các từ tương đồng về nghĩa sẽ được nhúng gần nhau trong một không gian đặc trưng có chiều thấp hơn (lower-dimensional feature space). Một thách thức lớn khác cũng đã được giải quyết nhờ kiến trúc sequence to sequence, cho phép ánh xạ chuỗi đầu vào thành chuỗi đầu ra với độ dài khác biệt. Kiến trúc này đặc biệt hiệu quả cho các bài toán phức tạp như Máy dịch, Hỏi đáp hay như Tạo chú thích video. Ưu điểm của nó là ít đưa ra giả định về cấu trúc chuỗi, đồng thời có thể xử lý tốt các cấu trúc câu đa dạng, dù ở thể chủ động hay bị động [1].



Hình 2.2: Cấu trúc chung của một mạng RNN

Trong giai đoạn khởi đầu của nghiên cứu Visual Question Answering, quy trình chung thường bao gồm hai bước tách bạch. Trước hết, một mạng tích chập chịu trách nhiệm trích xuất các đặc trưng thị giác từ ảnh và biến mỗi hình thành một low-dimensional vector. Tiếp theo, một mạng Long Short Term Memory (LSTM) được sử dụng để mã hóa câu hỏi thành một vectơ có ngữ cảnh, song song với biểu diễn ảnh. Hai vectơ này sau đó được ghép nối và đưa qua một hoặc hai lớp fully-connected nhằm dự đoán đáp án cuối cùng. Hai hạn chế chủ yếu của cách tiếp cận này là: Thứ nhất, cơ chế concatenate tại lớp fully-connected chỉ đơn thuần nối hai vectơ toàn cục, không có bất kỳ tương tác cục bộ nào giữa vùng ảnh quan trọng và cụm từ liên quan trong câu hỏi. Chẳng hạn, với câu "Có bao nhiêu quả bóng trên sân?", mô hình không thể tập trung vào các vùng chứa bóng mà chỉ tổng hợp chung toàn bộ thông tin ảnh. Thứ hai, LSTM chưa giải quyết được trước những câu hỏi đòi hỏi suy luận đa bước hoặc so sánh phức tạp, bởi toàn bộ ngữ cảnh dài hạn chỉ được nén lại trong một vectơ ẩn duy nhất trước khi đến lớp dự đoán.



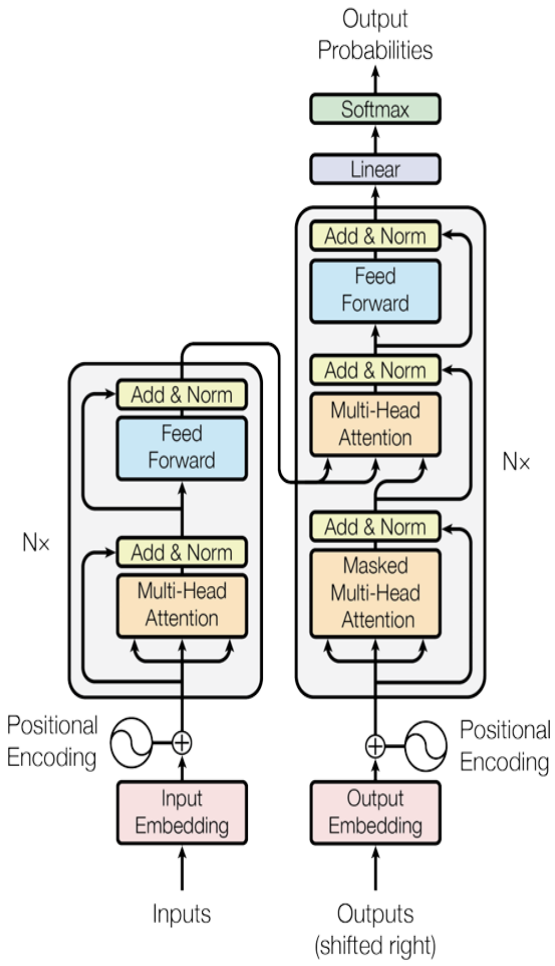
Hình 2.3: Sự kết hợp giữa CNN và LSTM

2.2. Kỷ nguyên của mô hình Transformer

Sau khi các mô hình CNN và LSTM lộ rõ hạn chế về khả năng tập trung và suy luận. Một State-of-the-art mang tính bước ngoặt là Attention, cho phép mô hình chủ động điều chỉnh sự tập trung, tương tự như cách con người tư duy. Nó giúp mô hình theo dõi một luồng thông tin nhất định và bỏ qua những chi tiết không liên quan đến tác vụ. Nhờ đó, kỹ thuật này đã chứng minh khả năng cải thiện đáng kể hiệu suất của các tác vụ như Máy dịch. Bằng cách cho phép decoder nhìn trực tiếp vào dữ liệu nguồn, nó không chỉ giải quyết được vấn đề nút thắt cổ chai (bottleneck) mà còn tạo ra một đường tắt đến các trạng thái ở xa, qua đó hỗ trợ khắc phục hiện tượng vanishing gradient. Một trong những kỹ thuật mô hình hóa dữ liệu chuỗi tân tiến nhất hiện nay là Transformers. Hoàn toàn dựa trên cơ chế attention, Transformers không yêu cầu xử lý dữ liệu đầu vào một cách tuần tự như mạng nơ-ron hồi quy. Điều này giúp các mô hình học sâu ghi nhớ ngữ cảnh từ những phần xa hơn trong các chuỗi dài tốt hơn. Hiện tại, đây là mô hình thống trị trong NLP và còn tối ưu hóa việc sử dụng GPUs nhờ khả năng xử lý song song. Các kiến trúc Transformer tiêu biểu như BERT, T5 hay GPT thường được pre-trained trên những tập dữ liệu khổng lồ và sau đó có thể được tinh chỉnh (fine-tuned) cho các tác vụ ngôn ngữ chuyên biệt. Chúng sở hữu khả năng ấn tượng trong việc tạo ra văn bản, thơ, mã nguồn và nhiều ứng dụng khác. Với những đột phá này, các mạng học sâu đã rất thành công trong việc trích xuất thông tin và tìm ra các biểu diễn ngữ nghĩa cho phương thức văn bản.

Các kiến trúc dựa trên Self-attention, đặc biệt là Transformers, đã trở thành lựa chọn hàng đầu trong xử lý ngôn ngữ tự nhiên. Chính từ những thành công vang dội này của NLP, nhiều nghiên cứu trong lĩnh vực thị giác máy tính đã tìm cách kết hợp các kiến trúc tương tự CNN với self-attention, một số khác thậm chí còn thay thế hoàn toàn các lớp tích chập. Tuy nhiên, những mô hình thuần self-attention này, dù rất tiềm năng về mặt lý thuyết, lại chưa được nhân rộng một cách hiệu quả trên các bộ tăng tốc phần cứng hiện đại do chúng thường đòi hỏi các pattern attention được thiết kế chuyên biệt. Học hỏi từ những thành công trong việc mở rộng quy mô của Transformer ở lĩnh vực NLP, một hướng thử nghiệm nổi bật là áp dụng trực tiếp một kiến trúc Transformer tiêu chuẩn cho dữ liệu hình ảnh. Do tính ứng dụng đa dạng của Computer Vision, các bài toán trong

lĩnh vực này rất phong phú và không ngừng là tâm điểm chú ý của nhiều công trình nghiên cứu. Với những bước tiến vũ bão của cả NLP và Computer Vision trong thời gian gần đây, việc hợp nhất hai phương thức này để giải quyết các bài toán đa phương thức chỉ còn là vấn đề thời gian. Sự tiến bộ vượt bậc trong lĩnh vực học sâu đa phương thức được minh chứng qua các mô hình gần đây, chẳng hạn như khả năng tạo sinh hình ảnh từ văn bản của DALL-E, Gemini hay GPT



Hình 2.4: Kiến trúc mạng Transformer

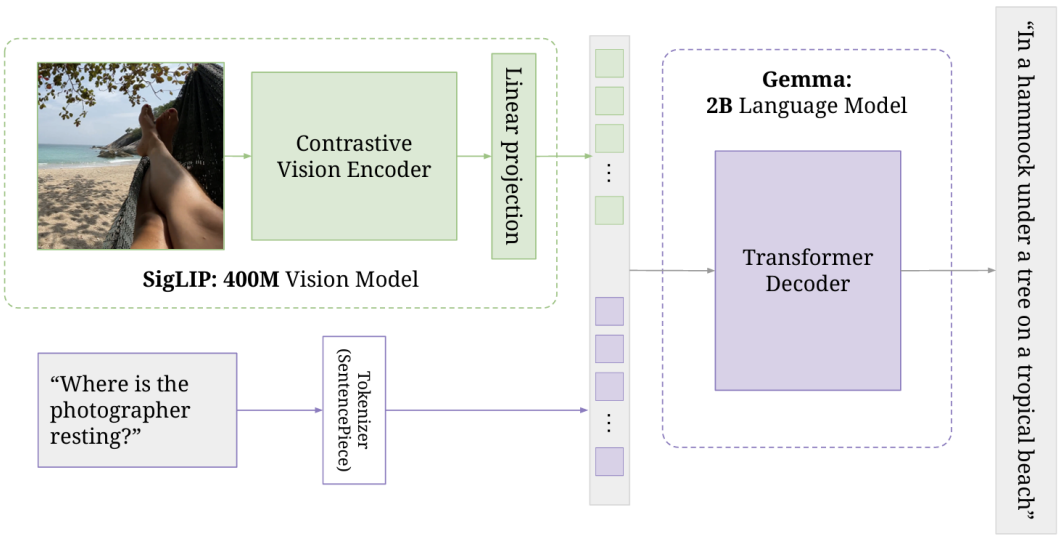
2.3. Các Mô hình Ngôn ngữ-Thị giác Lớn hiện đại

Sự thành công vang dội của các kiến trúc Transformer trong việc hợp nhất hai phương thức đã mở đường cho một thế hệ mô hình mới, vượt xa phạm vi của các tác vụ đơn lẻ: đó là các Mô hình Ngôn ngữ-Thị giác Lớn. Thay vì xây dựng các kiến trúc chuyên biệt từ đầu, hướng tiếp cận chủ đạo của nó là tận dụng sức mạnh của các Mô

hình Ngôn ngữ Lớn đã được tiền huấn luyện trên kho dữ liệu văn bản khổng lồ, và tìm cách dạy cho chúng khả năng nhìn. Triết lý này khai sinh ra kiến trúc Encoder-Decoder mạnh mẽ, trong đó một bộ mã hóa thị giác như Vision Transformer chịu trách nhiệm encode hình ảnh thành một chuỗi các image embedding, tương tự như một câu trong ngoại ngữ. Chuỗi nhúng này sau đó được xem như một tiền tố (prefix) cung cấp bối cảnh thị giác cho một bộ giải mã ngôn ngữ để nó có thể hiểu và tạo ra câu trả lời dưới dạng văn bản tự nhiên. Kiến trúc này đã đạt được những thành tựu đột phá với các mô hình tiêu biểu như Flamingo của DeepMind, vốn sử dụng cơ chế gated cross-attention để cho phép các lớp của LLM có thể nhìn vào các đặc trưng thị giác khi cần thiết. Một hướng đi khác là mô hình BLIP-2, mô hình này giới thiệu một thành phần cầu nối gọn nhẹ gọi là Q-Former để liên kết một Vision Encoder đã được frozen với một LLM có sẵn, giúp quá trình tiền huấn luyện trở nên cực kỳ hiệu quả về mặt tài nguyên. Nối tiếp thành công đó, các mô hình như PaLI-3 tiếp tục đẩy xa giới hạn bằng cách huấn luyện trên quy mô dữ liệu và số lượng tham số cực lớn, đạt được khả năng zero-shot và few-shot ấn tượng trên hàng loạt tác vụ đa phương thức [1].

Tuy nhiên, sức mạnh vượt trội của các mô hình Ngôn ngữ-Thị giác này đi kèm với một thách thức cố hữu: quy mô khổng lồ. Các mô hình với hàng chục, thậm chí hàng trăm tỷ tham số, được tiền huấn luyện trên hàng tỷ cặp ảnh-văn bản đòi hỏi chi phí tính toán cực lớn, tạo ra một rào cản đáng kể cho cộng đồng nghiên cứu và việc triển khai vào ứng dụng thực tế. Sự cồng kềnh này không chỉ khiến việc tinh chỉnh (fine-tuning) cho các tác vụ cụ thể trở nên khó khăn mà còn hạn chế khả năng ứng dụng trên các thiết bị có tài nguyên giới hạn. Chính từ bài toán về sự cân bằng giữa hiệu năng và hiệu quả này, một hướng nghiên cứu mới đã hình thành: phát triển các VLM có kiến trúc thông minh và gọn nhẹ hơn nhưng vẫn kế thừa được sức mạnh từ các mô hình tiền bối. Đây là bối cảnh ra đời của PaliGemma, một kiến trúc kế thừa triết lý Encoder-Decoder hiệu quả từ họ mô hình PaLI nhưng được xây dựng hoàn toàn dựa trên các thành phần mã nguồn mở và đã được tối ưu về kích thước là SigLIP và Gemma.

2.4. Kiến trúc của mô hình PaliGemma



Hình 2.5: Kiến trúc của PaliGemma

PaliGemma là một mô hình ngôn ngữ thị giác, lấy cảm hứng từ PaLI-3 và dựa trên các thành phần như mô hình thị giác SigLIP (Sigmoid Loss for Language Image Pre-Training) và mô hình ngôn ngữ Gemma. PaliGemma sử dụng cả hình ảnh và văn bản làm dữ liệu đầu vào, đồng thời có thể trả lời các câu hỏi về hình ảnh một cách chi tiết và theo bối cảnh. Điều này có nghĩa là PaliGemma có thể phân tích được hình ảnh sâu hơn và cung cấp thông tin chi tiết hữu ích, chẳng hạn như chú thích cho hình ảnh và video ngắn, phát hiện đối tượng và đọc văn bản được nhúng trong hình ảnh.

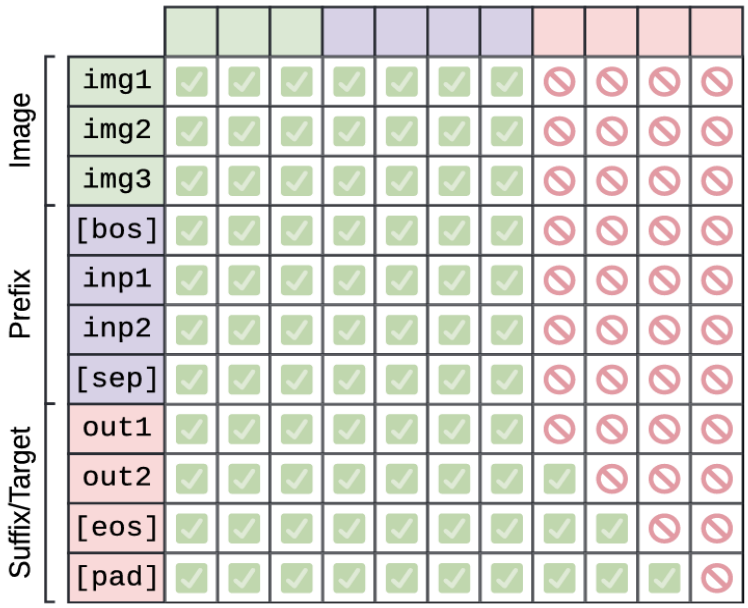
Về mặt tổng quan kiến trúc, PaliGemma gồm 3 thành phần chính [2]:

1. Một bộ encoder hình ảnh, ở đây phía bên Google DeepMind sử dụng là một checkpoint SigLIP được open-source, cụ thể là phiên bản ViT So400M được tối ưu hóa hình dạng (shape optimized). Mô hình này đã trải qua quá trình tiền huấn luyện tương phản (contrastively pretrained) trên quy mô lớn sử dụng hàm mát sigmoid
2. Một mô hình ngôn ngữ chỉ có decoder (decoder-only language model), ở phía Google DeepMind chọn là checkpoint mô hình open-source tiền huấn luyện thô (raw pre-trained checkpoint) Gemma-2B v1.0.

3. Một lớp tuyến tính (linear layer) thực hiện phép chiếu (projecting) các token đầu ra của SigLIP về cùng một chiều (dimension) với các token từ vựng (vocab tokens) của Gemma 2B, để chúng có thể được nối kết (concatenated).

Hình ảnh được đưa qua image encoder, biến đổi nó thành một chuỗi gồm các token hình ảnh. văn bản được chuyển đổi thành các token văn bản sử dụng bộ mã hóa token SentencePiece của Gemma, và chúng được nhúng (embedded) bởi lớp nhúng từ vựng (vocabulary embedding layer) của Gemma. Các token hình ảnh được chiếu (projected) bằng lớp chiều tuyến tính (linear projection). Sau đó, chuỗi các token đầu vào cho bộ decoder được tạo như sau: tokens = [image tokens, ..., BOS (begin of sentence), prefix tokens, ..., SEP (separator), suffix tokens, ..., EOS (end of sentence), PAD (padding)...].

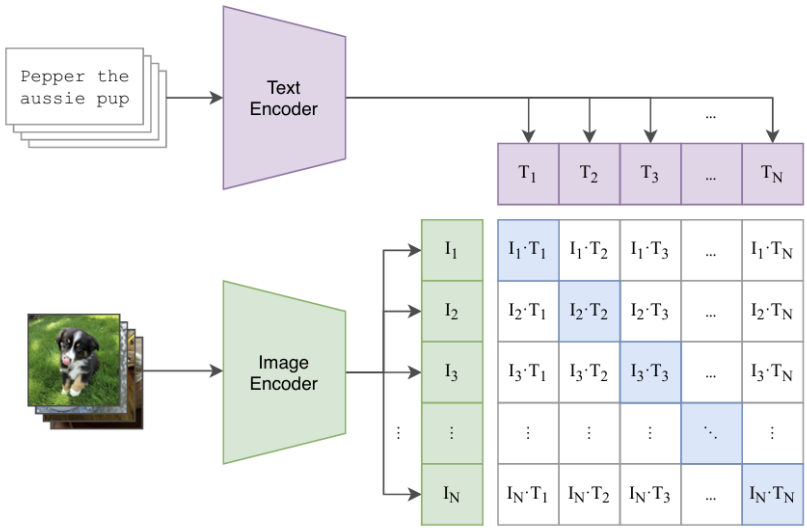
Hình ảnh luôn được đưa về một kích thước cố định (224x224, 448x448, 896x896 pixel). Điều này dẫn đến một số lượng token hình ảnh cố định cho mỗi biến thể mô hình (tương ứng là 256, 1024 hoặc 4096 token) mà phía Google DeepMind đặt ở phía trước, giúp các token hình ảnh dễ dàng được diễn giải mà không cần đến các đánh dấu vị trí đặc biệt (special location markers). Token BOS sau đó đánh dấu sự bắt đầu của các token văn bản. "\n" thường được làm token SEP, nó không xuất hiện trong bất kỳ tiền tố (prefix) nào. Phía Google DeepMind cũng mã hóa token SEP riêng biệt để tránh nó bị gộp với phần cuối của tiền tố hoặc phần đầu của hậu tố (suffix). Để tối đa hóa khả năng của mô hình đối với một mô hình nhỏ như vậy, họ áp dụng cơ chế chú ý đầy đủ (full attention) (không che giấu - unmasked) trên toàn bộ đầu vào, tức là các token hình ảnh và token tiền tố. Bằng cách này, các token hình ảnh cũng có thể nhìn về phía trước (look ahead) vào nhiệm vụ hiện tại để cập nhật biểu diễn của chúng. Phần hậu tố đóng vai trò là đầu ra của mô hình và nó bắt buộc phải được xử lý dưới mặt nạ tự hồi quy (auto-regressive mask), áp dụng cho cả các token PAD. Về độ dài chuỗi, họ quy ước đó là tổng số token trong tiền tố (prefix) và hậu tố (suffix), không tính token hình ảnh.



Hình 2.6: Cơ chế Prefix-LM masking của PaliGemma

2.5. Sigmoid Loss for Language Image Pre-Training

Thành phần mã hóa thị giác được lựa chọn trong mô hình đề xuất là một checkpoint của SigLIP, một kiến trúc được tối ưu hóa dựa trên nền tảng của Vision Transformer. Để hiểu rõ lý do lựa chọn và ưu điểm của SigLIP, trước hết cần phân tích mô hình tiền thân của nó là Contrastive Language-Image Pre-Training (CLIP) của OpenAI.



Hình 2.7: Mô hình CLIP

Đầu tiên, có một tập dữ liệu gồm các cặp hình ảnh và văn bản tương ứng. Trong quá trình huấn luyện, một lô dữ liệu (batch) các cặp này được lấy ra. Các hình ảnh trong batch được đưa qua Image Encoder, kết quả là ta thu được một vector biểu diễn (representation vector) duy nhất cho mỗi hình ảnh. Tương tự, các văn bản tương ứng trong batch được đưa qua Text Encoder, và ta cũng thu được một vector biểu diễn duy nhất cho mỗi văn bản. Tiếp theo, để đo lường sự tương đồng (matching) giữa hình ảnh và văn bản, ta tính tích vô hướng giữa vector biểu diễn của mỗi hình ảnh với vector biểu diễn của mỗi văn bản trong cùng một batch. Điều này tạo ra một ma trận tương đồng như đường chéo trong hình 3, ngược lại đối với các cặp không tương xứng sẽ cho tích vô hướng có giá trị thấp. Giờ ta sẽ xây dựng hàm Entropy loss để "ép" các giá trị matching cao nổi bật hơn và đẩy giá trị của các cặp không tương xứng xuống.

Nhưng có một vấn đề với CLIP là nó sử dụng hàm softmax, khi batch chứa nhiều cặp ảnh và văn bản, tổng mẫu số của softmax càng lớn, dẫn đến việc xác suất của mỗi cặp bị phân bổ mỏng đi. Hệ quả là mô hình chỉ tập trung tối ưu cho cặp có matching cao nhất, còn những cặp đúng khác bị xem như negative, dẫn đến hiện tượng false negative giữa các cặp trong batch:

$$-\frac{1}{2|\beta|}\sum_{i=1}^{|\beta|}\left(log\frac{e^{tx_i\cdot y_i}}{\Sigma_{j=1}^{|\beta|}\ e^{tx_i\cdot y_j}}+log\frac{e^{tx_i\cdot y_i}}{\Sigma_{j=1}^{|\beta|}\ e^{tx_j\cdot y_i}}\right)$$

Trong đó $x_i=\frac{f(I_i)}{||f(I_i)||_2}$ và $y_i=\frac{g(T_i)}{||g(T_i)||_2}$. Từ công thức có thể dễ dàng nhận thấy với mỗi $|\beta|$ cặp (I_i, T_i), ta phải tính song song cả hai phần loss $image \rightarrow text\ softmax$ và $text \rightarrow image\ softmax$. [3]

Để khắc phục những thách thức của hàm mất mát Softmax, paper "Sigmoid Loss for Language Image Pre-Training" của Zhai và cộng sự ra mắt năm 2023\ đã đề xuất một phương pháp tiếp cận khác biệt. Thay vì định hình bài toán như một bài toán phân loại đa lớp trên toàn bộ batch, SigLIP tái cấu trúc nó thành một tập hợp các bài toán phân loại nhị phân độc lập. Mỗi cặp ảnh và văn bản được đánh giá riêng lẻ để trả lời câu hỏi: "Cặp này có tương ứng với nhau hay không?".

Cách tiếp cận này được hiện thực hóa bằng cách áp dụng hàm sigmoid trực tiếp lên điểm tương đồng của từng cặp. Hàm mất mát được xây dựng dựa trên binary cross-entropy, tối ưu hóa trên tất cả các cặp trong batch. Điểm cốt lõi là khi ta dùng sigmoid trong SigLIP, ta chỉ cần tính loss một lần cho mỗi cặp, không cần chuẩn hóa trên toàn bộ các cặp khác:

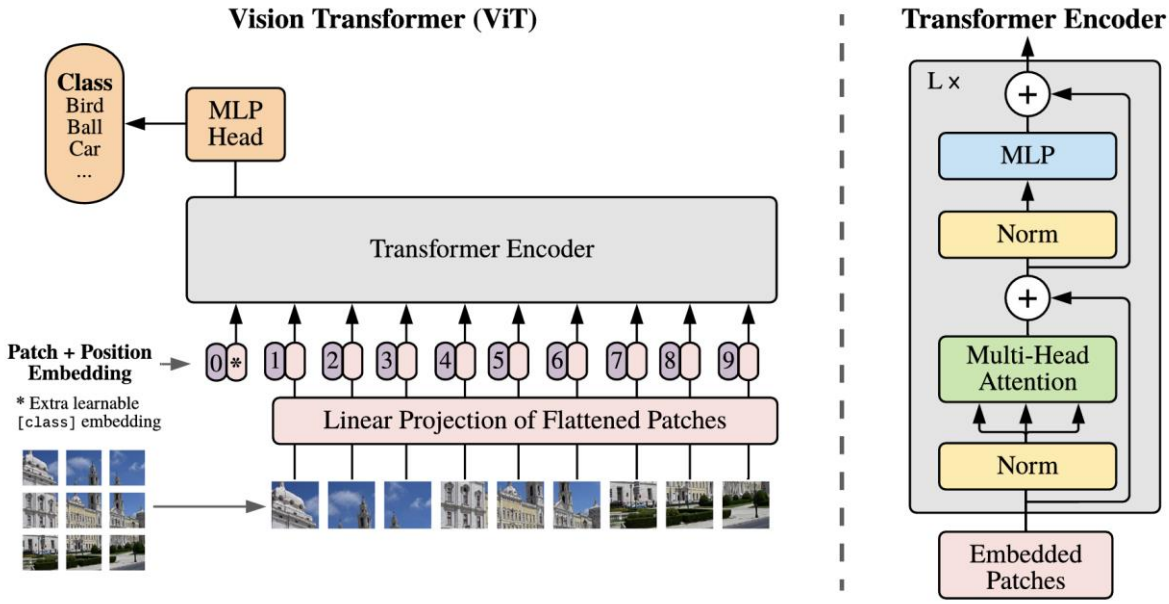
$$-\frac{1}{|\beta|}\sum_{i=1}^{|\beta|}\sum_{j=1}^{|\beta|} log\frac{1}{1+e^{z_{ij}(-tx_i\cdot y_i+b)}}$$

Trong đó z_{ij} là nhãn cho cặp hình ảnh và văn bản cho trước, có giá trị bằng 1 nếu chúng matching với nhau và bằng -1 trong trường hợp ngược lại [4]. Cách tiếp cận này mang lại nhiều ưu điểm vượt trội so với phương pháp của CLIP. Thứ nhất Độc lập với Batch do mỗi cặp được xử lý độc lập, hàm mất mát không còn phụ thuộc vào các cặp âm tính khác trong batch. Điều này loại bỏ hoàn toàn vấn đề "false negative" ở trên, cho phép mô hình học được các mối quan hệ ngữ nghĩa tinh tế hơn. Thứ hai hiệu quả về tài nguyên do việc loại bỏ bước chuẩn hóa trên toàn batch giúp giảm đáng kể chi phí tính toán, cho phép SigLIP huấn luyện hiệu quả với các kích thước batch đa dạng, kể cả các batch rất lớn hoặc rất nhỏ, mà không làm suy giảm đáng kể hiệu năng. Thứ ba cải thiện chất lượng biểu diễn: Bằng cách cho phép nhiều cặp văn bản có ngữ nghĩa khác nhau cùng tương ứng với một hình ảnh, mô hình có khả năng học được một không gian nhúng phong phú và linh hoạt hơn.

2.5.1 Bộ Image Encoder Vision Transformer

Khái niệm này lần đầu tiên được nói đến tại Hội nghị Quốc tế về Biểu diễn Học tập (International Conference on Learning Representations) năm 2021. Paper có tên "An Image is Worth 16x16 Words: Transformers for Image Recognition at Scale". Về cơ bản mô hình Transformer là mô hình Sequence to Sequence nghĩa là ta đưa một chuỗi các embedding đầu vào và đầu ra là chuỗi các embedding được ngữ cảnh hóa, còn với Vision Transformer (Encoder-only) ta encode một hình ảnh bằng cách chia nó thành các patches 16x16 hoặc có thể là 32x32, 9x9. Mỗi patch tương ứng với một vùng không gian xác định trên ảnh gốc. Sau đó được sắp xếp thành một chuỗi để làm đầu vào cho bộ mã hóa Transformer [5], ví dụ nếu ta chia ảnh thành các patches 16x16, patch 1 sẽ

biểu biểu nhóm các điểm ảnh ở trên cùng bên trái, patch 4 sẽ biểu diễn trên cùng bên phải, patch 13 sẽ là dưới cùng bên trái còn patch 16 sẽ là dưới cùng bên phải. Ta trích xuất thông tin của mỗi patch bằng cách sử dụng tích chập (convolution) sau đó ta làm phẳng nó (flatten) để nó trở thành một chuỗi các patch thay vì là một grid được chia như ban đầu, điều này sẽ làm mất đi thông tin về vị trí trước khi được tích chập. Đó là lý do ta phải thêm vào Position Encoder để nói cho mô hình biết vị trí ban đầu của các patch. Không giống như mô hình Transformer gốc hoặc các mô hình Transformer thường thấy cho NLP, các position encoding không được tính toán bằng cách sử dụng sinusoidal function mà chúng là các vector học được. Do đó, với mỗi vị trí xác định trong chuỗi đầu vào, mô hình sẽ có một position encoder vector riêng biệt được học. Những vector này luôn được cộng thêm vào vector embedding của patch ảnh tương ứng tại vị trí đó trong chuỗi. Ví dụ, vector mã hóa vị trí cho vị trí đầu tiên trong chuỗi sẽ luôn được cộng vào embedding của patch được trích xuất từ góc trên bên trái của ảnh. Sau đó ta sẽ cho chúng vào Transformer Encoder.



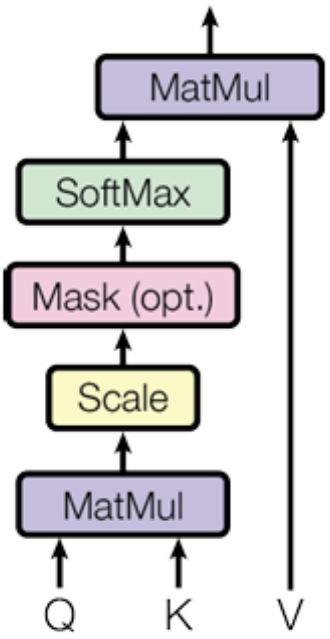
Hình 2.8: Mô hình Vision Transformer

- Single-Head Attention:

Đầu vào của một Attention Block gồm N embeddings với D chiều, N ở đây là số patch + 1 (Class token), việc đầu tiên ta làm là stack các embedding thành một ma trận $X \in$

$R^{N\times D}$. Bài toán mà Attention giải quyết là làm sao để N embeddings giao tiếp được với nhau nhằm chia sẻ thông tin. Ta chiếu mỗi patch embedding ba quá trình riêng biệt để tạo ra Q (Queries), K (Keys) và V (Values). Queries đại diện cho "Biểu diễn phần tử cần tính Attention" $W^Q \in R^{D\times d_k}$, Keys đại diện cho "Biểu diễn các phần tử để so sánh sự liên quan" $W^K \in R^{D\times d_k}$, Values đại diện cho "Nội dung thông tin của các phần tử" $W^V \in R^{D\times d_v}$, Trong đó d_k là chiều của Queries và Keys, d_v là chiều của Values. sau đó ta có thể tính Q,K và V bằng cách nhân với ma trận embedding gốc X. $Q = XW^Q \in R^{N\times d_k}$, $K = XW^k \in R^{N\times d_k}$ và $v = XW^v \in R^{N\times d_v}$. Giờ đây ta có thể tính toán cơ chế cốt lõi của Attention gọi là Scaled Dot-Product Attention:

$$Y = softmax\left(\frac{QK^T}{\sqrt{d_k}}\right)V \in R^{N\times d_v} \text{ [6]}$$



Hình 2.9: Cơ chế Scale dot product

- Multi-Head Attention:

Vậy nếu các patch muốn gửi nhiều thông tin đi thì sao? Giải pháp này được nói trong paper gốc Attention Is All You Need là ta lại đi tính nhiều phép tính attention song song, sử dụng một tập hợp gồm các Head attention điều này cho phép mô hình đồng thời chú

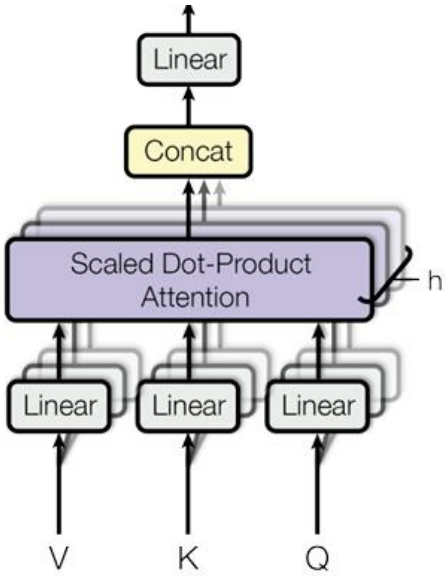
ý đến thông tin từ các không gian con biểu diễn khác nhau tại các vị trí khác nhau. Sau khi thực hiện phép lặp từng Head:

for h = 1,2...,H:

$$Q_h = XW_h^Q$$
$$K_h = XW_h^K$$
$$V_h = XW_h^V$$
$$Head_h = softmax\left(\frac{Q_h K_h^T}{\sqrt{d_k}}\right) V_h$$

Ta nối các kết quả lại với nhau và thực thực phép chiếu cuối W^O để thu được dimension mà ta mong muốn:

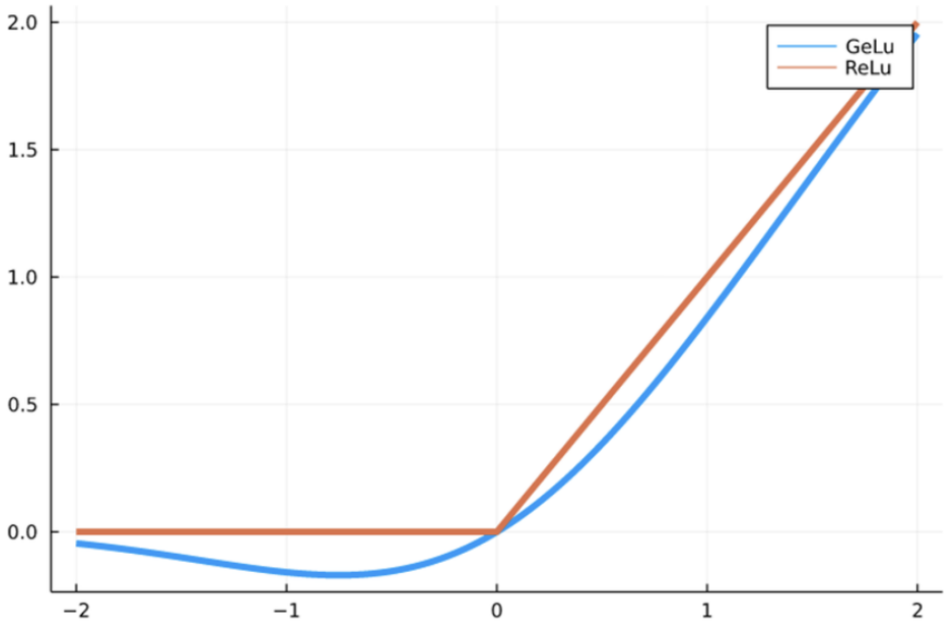
$$MultiHead(X) = Concat(head_1, head_2, \ldots, head_h)W^O \text{ [6]}$$



Hình 2.10: Cơ chế Multi-Head Attention

- Multi-Layer Perceptron (MLP):

Sau khi các embedding đã chia sẻ "suy nghĩ" với nhau, ta lại bắt chúng "suy nghĩ" một mình những kiến thức mà chúng học được. Ta khởi tạo một MLP gồm 2 lớp trên mỗi embedding $MLP(x) = W_2\sigma(W_1 x + b_1) + b_2$, ở đây $\sigma(\quad)$ là hàm kích hoạt phi tuyến tính, ở trong paper gốc họ dùng ReLU hay gần đây được sử dụng nhiều là GeLU



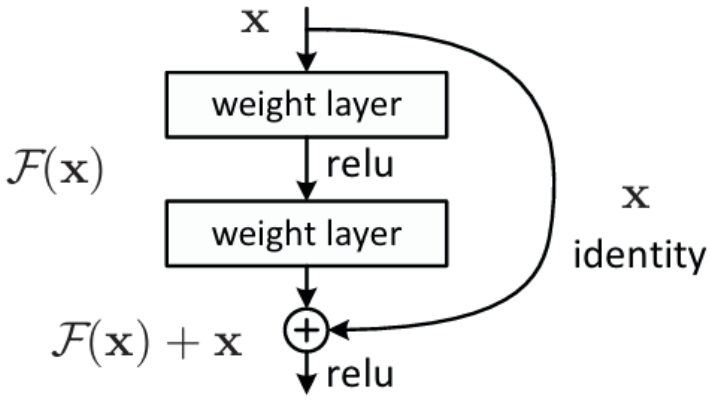
Hình 2.11: Hàm kích hoạt ReLU và GeLU

Residual Connection:

Lần đầu xuất hiện trong paper Deep Residual Learning for Image Recognition năm 2015, nhằm mục đích giảm thiểu vấn đề vanishing gradient. Trong quá trình lan truyền ngược (back-propagation), gradient được nhân với đạo hàm của hàm kích hoạt. Với trường hợp hàm ReLU, điều này có nghĩa là trong khoảng một nửa các trường hợp, gradient sẽ bằng không. Nếu không có cơ chế kết nối tắt (skip connection), một phần lớn tín hiệu huấn luyện (gradient) sẽ bị mất đi trong quá trình lan truyền ngược khi đi qua nhiều lớp. Các skip connection làm giảm bớt tác động này, bởi vì phép cộng là tuyến tính đối với đạo hàm. Do đó, mỗi khối residual cũng nhận được một gradient đi thẳng qua skip connection mà không bị ảnh hưởng bởi các đạo hàm có thể bằng không hoặc rất nhỏ. Các phép cộng trong các skip connection tạo thành một đường dẫn trong đồ thị tính toán, nơi mà gradient không bị mất đi [7].

Một tác dụng khác của các skip connection là thông tin vẫn giữ được tính cục bộ (local) ở một mức độ nào đó trong các lớp Transformer. Cơ chế self-attention cho phép luồng thông tin tùy ý trong mạng và do đó cho phép mối quan hệ giữa các token ở bất kỳ vị trí nào. Tuy nhiên, các skip connection luôn "nhắc nhở" biểu diễn về trạng thái ban đầu là

gì. Ở một mức độ nào đó, các skip connection đảm bảo rằng các biểu diễn theo ngữ cảnh của các token đầu vào vẫn thực sự biểu diễn các token gốc.



Hình 2.12: Residual learning

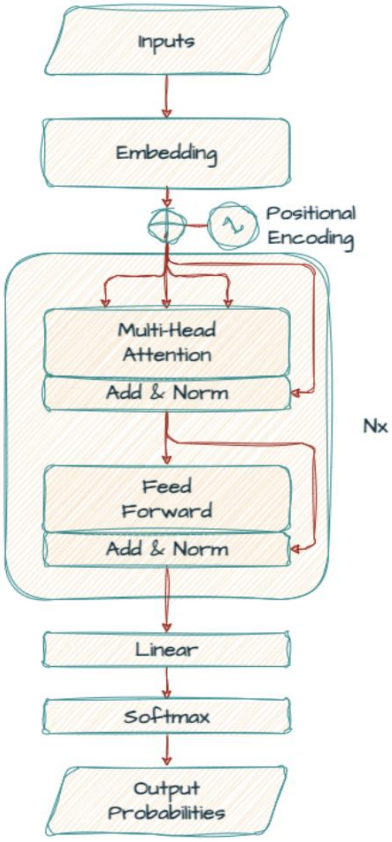
- Layer normalization:

Giờ ta sẽ đi vào phần cuối cùng nằm trong khối Encoder, hai khối layer norm xuất hiện trước khối Multi-Head Attention và khối MLP. Layer norm khá giống với batch norm ở chỗ ta đều chuẩn hóa input bằng cách trừ đi giá trị trung bình (mean) của chúng và chia cho độ lệch chuẩn có bổ sung một hằng số bù dương nhỏ để đảm bảo ổn định số học. Để tránh làm mất khả năng biểu đạt của dữ liệu, kết quả này được điều chỉnh tỷ lệ bằng một tham số độ lợi (gain parameter) học được và cộng thêm một tham số độ lệch (bias parameter) học được: $LayerNorm(x)_i = \gamma_i \frac{x-\mu}{\sqrt{\sigma^2+\epsilon}} + \beta_i$ [8]. Các kỹ thuật chuẩn hóa này được sinh ra để xử lý covariate shift, vấn đề này xảy ra khi input layer thay đổi độ lớn output của layer quá nhiều mà bởi vì loss của mô hình khi huấn luyện dựa trên output cho nên loss cũng sẽ thay đổi rất nhiều điều này ảnh hưởng đến quá trình tính toán gradient trong quá trình lan truyền ngược (backpropagation) và do gradient xác định cách chúng ta cập nhập trọng số (weight) của mô hình trong quá trình huấn luyện nên các bộ trọng số này cũng sẽ thay đổi rất nhiều dẫn đến mô hình học rất chậm.

2.5.2. Bộ Language Decoder Gemma

Gemma là một mô hình ngôn ngữ lớn open-source. Kiến trúc của Gemma được phát triển và cải tiến dựa trên kiến trúc LLM của paper Attention Is All You Need được nhóm

Google Research đưa ra vào năm 2017. Vì cấu trúc là decoder-only nên chức năng chính của Gemma là tạo ra văn bản từng từ một (token-by-token) dựa trên prompt mà người dùng cung cấp.



Hình 2.13: Cấu trúc của Gemma dựa trên Transformer decoder

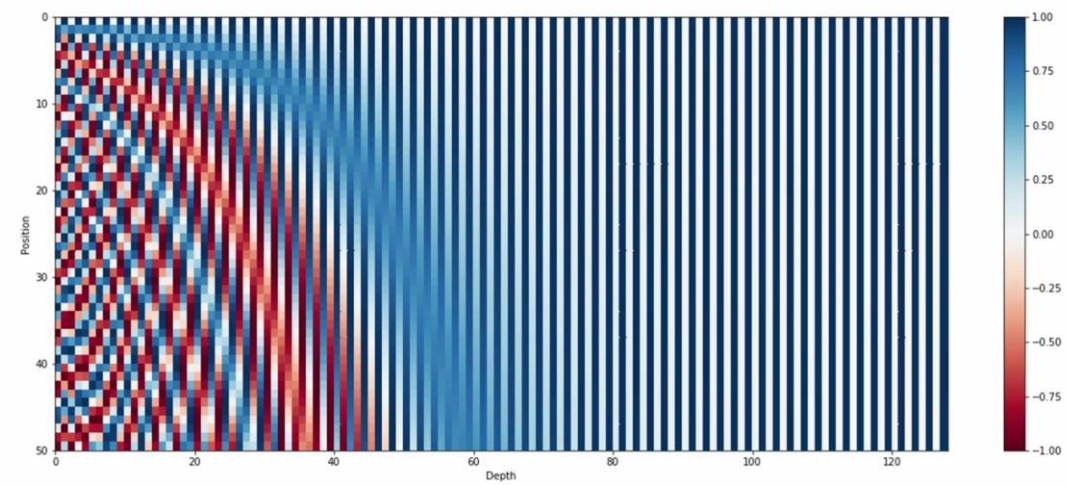
- Multi-Query Attention (MQA):

Khác với Multi-Head Attention ở block encoder của ViT trong mục trước. MQA lấy các head của các Query nhân với tích vô hướng với chỉ một Key rồi lấy softmax của chúng nhân với chỉ một Key còn concat chúng lại với nhau. Điều này làm tăng đáng kể tốc độ tính toán của decoder gấp nhiều lần so với Multi-Head Attention nhưng đồng thời cũng làm tăng chỉ số perplexity của mô hình lên một chút. Trong paper gốc “Fast Transformer Decoding: One Write-Head is All You Need” của Noam Shazeer có chỉ ra tốc độ decoder giảm từ 203 microsecond xuống còn 33 microsecond và perplexity tăng nhẹ thành 30,2 so với 29,9 của Multi-Head Attention [9]. Ở đây, chỉ số perplexity đo lường mức độ "bối rối" của mô hình khi dự đoán từ tiếp theo, giá trị càng thấp càng tốt. Mức

tăng chỉ 0.3 điểm (từ 29.9 lên 30.2) được xem là không đáng kể, trong khi tốc độ tính toán lại được cải thiện tới 6 lần. Đây là một sự đánh đổi cực kỳ hiệu quả, cho phép mô hình tạo văn bản nhanh hơn nhiều mà gần như không suy giảm về chất lượng ngôn ngữ.

- Rotary Position Embeddings (RoPE):

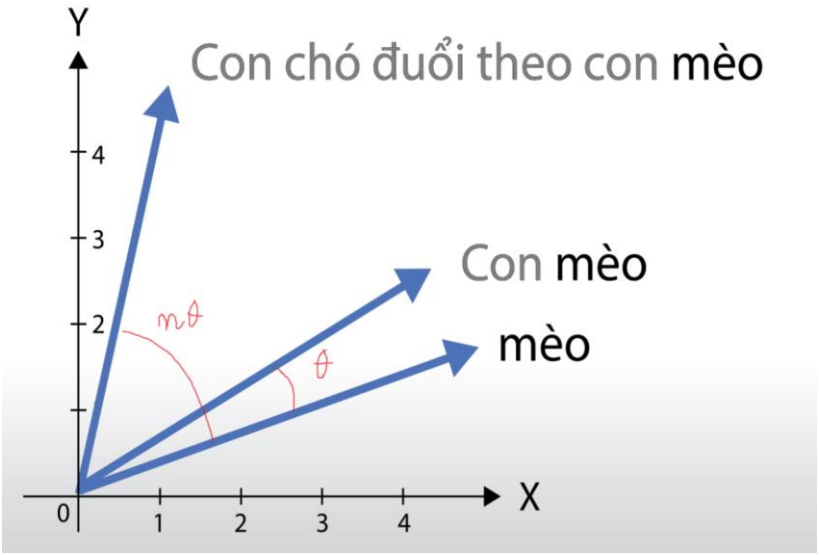
Khác với Position encoder trong Vision Transformer sử dụng các learnable positional embedding, trong paper gốc của Transformer phía Google giới thiệu phương pháp sinusoidal positional encoding trong đó mỗi vị trí được gán một vector đặc trưng duy nhất, được xây dựng từ các thành phần tuần hoàn với các tần số khác nhau: $PE_{(pos,2i)} = sin(pos/10000^{2i/d_{model}})$ cho các Position Embedding ở vị trí chẵn và $PE_{(pos,2i+1)} = cos(pos/10000^{2i/d_{model}})$ cho các Position Embedding ở vị trí lẻ [6].



Hình 2.14: Sinusoidal Positional Encoding

Nhưng với mô hình Gemma, bên phía Google sử dụng Rotary Position Embeddings. Ý tưởng chính của nó là thay vì thêm một positional để nhúng vào vector để mã hóa vị trí của từ trong câu, họ đề xuất áp dụng một phép quay lên vector. Cho ví dụ có một vector từ hai chiều đại diện cho từ "mèo". Khi đó, để biểu diễn từ "mèo" xuất hiện ở vị trí thứ hai trong một câu, ta thực hiện một phép quay lên vector đó. Giả sử góc quay đó là θ. Nếu từ đó xuất hiện ở một vị trí xa hơn nữa trong câu, thì góc xoay của vector đó sẽ lớn hơn và góc mà ta xoay là một bội số nguyên nθ của vị trí từ đó trong câu, điều này có lợi hơn nhiều so với sử dụng Absolute Positional Embeddings ở trên bởi vì nếu ta có

thêm token vào sau từ "mèo", vector cho phần đầu câu sẽ vẫn giữ nguyên cho dù ta thêm rất nhiều token đi nữa thì vẫn sẽ không bị ảnh hưởng.



Hình 2.15: Vector truy vấn hai chiều cho token mèo

Ưu điểm chính của RoPE là nó giúp bảo toàn vị trí tương đối giữa các từ. Điều này có nghĩa là, khi xét hai từ "mèo" và chó" có cùng một khoảng cách tương đối trong câu, thì giá trị tích vô hướng giữa vector biểu diễn của chúng sẽ giữ nguyên, bất kể vị trí tuyệt đối của chúng trong câu là gì thì nó vẫn sẽ giữ một góc là θ. Sự đánh đổi là một độ phức tạp cao hơn một chút trong quá trình triển khai, nhưng lợi ích về hiệu năng mà nó mang lại cho các tác vụ suy luận phức tạp đã khiến nó trở thành tiêu chuẩn trong các LLM hiện đại như Llama và Gemma. Đây là công thức tính Rotary Embeddings cho trường hợp 2 dimension:

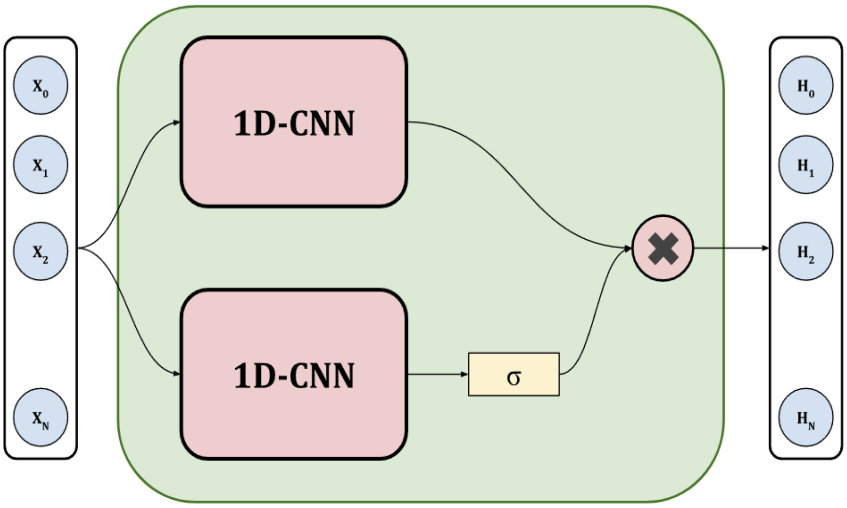
$$f_{\{q,k\}}(x_m, m) = \begin{pmatrix} cos\, m\theta & -sin\, m\theta \\ sin\, m\theta & cos\, m\theta \end{pmatrix} \begin{pmatrix} W_{\{q,k\}}^{(11)} & W_{\{q,k\}}^{(12)} \\ W_{\{q,k\}}^{(21)} & W_{\{q,k\}}^{(22)} \end{pmatrix} \begin{pmatrix} x_m{}^{(1)} \\ x_m{}^{(2)} \end{pmatrix} \quad [10]$$

Ở đây $\begin{pmatrix} cos\, m\theta & -sin\, m\theta \\ sin\, m\theta & cos\, m\theta \end{pmatrix}$ là phép quay ma trận một góc $m\theta$ (m là absolute positional của token trong câu), x là vector chúng ta quay cuối cùng là $\begin{pmatrix} W_{\{q,k\}}^{(11)} & W_{\{q,k\}}^{(12)} \\ W_{\{q,k\}}^{(21)} & W_{\{q,k\}}^{(22)} \end{pmatrix}$ phép biến đổi tuyến tính để lấy query và key vector trước khi áp dụng ma trận quay. Còn đây là công thức tổng quát khi có nhiều hơn 2 dimension:

$$f_{\{q,k\}}(x_m, m) = R^d_{\Theta,m} W_{\{q,k\}} x_m \text{ [10]}$$

Gaussian Error Gated Linear Unit (GeGLU):

Trước tiên, ta hãy tìm hiểu Gated Linear Unit (GLU). Được giới thiệu trong paper "Language Modeling with Gated Convolutional Networks" vào năm 2017. GLU được thiết kế nhằm nâng cao khả năng mô hình hóa của mạng neural thông qua việc tích hợp một cơ chế cổng (gating) để điều tiết luồng các feature đầu vào. Về cơ bản, GLU phân chia đầu vào thành hai nhánh: một nhánh được biến đổi tuyến tính, trong khi nhánh còn lại được đưa qua một hàm kích hoạt sigmoid để hoạt động như một cổng. Đầu ra của hai phép biến đổi này sau đó được nhân từng phần tử, cho phép kiểm soát lượng features được truyền đi. Cơ chế cổng này đóng vai trò quan trọng trong việc giúp mô hình tập trung vào các feature phù hợp và bỏ qua dữ liệu không liên quan, từ đó thúc đẩy quá trình học hiệu quả hơn và cải thiện hiệu suất, đặc biệt là trong các tác vụ xử lý ngôn ngữ tự nhiên và mô hình chuỗi. Bằng cách lọc thông tin một cách chọn lọc, GLU còn góp phần giải quyết vanishing gradient và tăng tốc độ hội tụ của các mạng deep neural [11]. Yann Dauphin và đội ngũ của ông là những người đóng góp then chốt vào sự phát triển và phổ biến của Gated Linear Unit thông qua bài báo có tầm ảnh hưởng năm 2017 của họ. Nghiên cứu của họ không chỉ làm nổi bật hiệu quả của GLU trong việc cải thiện mô hình hóa ngôn ngữ mà còn truyền cảm hứng cho việc khám phá các biến thể của nó cho các mô hình ngôn ngữ lớn sau này như SwiGLU cho Llama hay GeGLU cho Gemma.



Hình 2.16: Gated Linear Unit

Quay trở lại với mục chính. GeGLU ở đây đơn giản chỉ là biến thể của GLU và được đề cập trong paper GLU Variants Improve Transformer của Noam Shazeer. Như RLU, GeGLU cũng chia làm hai phần: một phần có dạng GELU và một phần tuyến tính. Đầu ra của dạng GELU được nhân từng phần tử với phần tuyến tính, tạo ra một hàm kích hoạt phi tuyến (non-linear):

$$GLU = (x, W, V, b, c) = \sigma(xW + b) \otimes (xV + c)$$
$$GeGLU = (x, W, V, b, c) = GELU(xW + b) \otimes (xV + c)\ [12]$$

- Root Mean Square Normalization (RMSNorm):

Trong mục trước, ta đã biết Layer Norm giúp model ổn định quá trình huấn luyện và thúc đẩy sự hội tụ. Điều này nhờ khả năng xử lý việc tái định tâm (re-centering) và tái tỉ lệ (re-scaling) của cả đầu vào lẫn ma trận trọng số. Tuy nhiên, trong paper Root Mean Square Layer Normalization của Biao Zhang và Rico Sennrich lại đưa ra giả thuyết rằng trong tác vụ NLP việc re-centering là không cần thiết, việc này làm tăng chi phí tính toán khiến quá trình trình toán qua các layer bị chậm đi. Bằng cách loại bỏ phép trừ trung bình μ đi: $RMSNorm(x)_i = \gamma_i \frac{x_i}{RMS}$ trong đó $RMS = \sqrt{\frac{1}{d}\sum_{i=1}^{d} \quad x_i^2 + \varepsilon}$ [13]. Giúp làm đơn giản hóa luồng dữ liệu, chỉ cần duyệt để lấy bình phương một lần để chuẩn hóa thay vì hai lần như Layer Norm.

2.6. Lớp ánh xạ tuyến tính

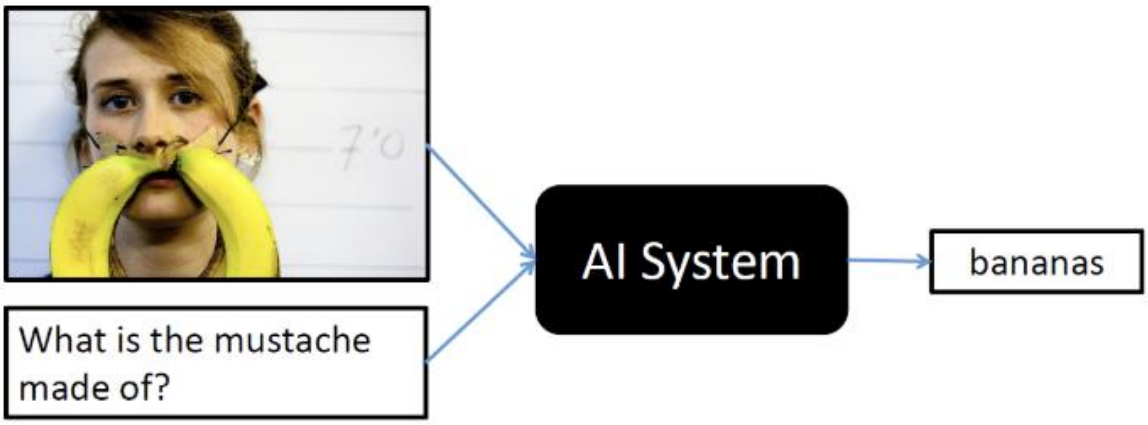
Lớp này là một mạng nơ ron cơ bản gồm một lớp duy nhất có nhiệm vụ chuyển đổi các image embedding biểu diễn hình ảnh từ Siglip sang định dạng tương thích với các word embedding của Gemma: $y = W \cdot x + b$, trong đó y là Vector đầu ra đã được ánh xạ có kích thước d_{Gemma}, W là ma trận trọng số có kích thước $d_{Gemma} \times d_{Siglip}$, x là vector image embedding đầu vào có kích thước d_{SigLip}, b là vector bias có kích thước d_{Gemma}. Các tham số W và b được học trong quá trình training thông qua quá trình backpropagation. Bằng cách này, Lớp kết nối hoạt động như một bộ chuyển đổi, đảm bảo rằng thông tin thị giác có thể được kết nối với thông tin văn bản để tạo thành chuỗi đầu vào hoàn chỉnh cho bộ giải mã. Điều này cho phép Gemma hiểu và xử lý đồng thời cả hai phương thức dữ liệu một cách hiệu quả.

CHƯƠNG 3: Xây dựng Mô hình hỏi đáp dựa trên hình ảnh

Chương này trình bày chi tiết về quá trình thiết lập môi trường, huấn luyện và đánh giá hiệu năng của mô hình PaliGemma đã được đề xuất ở Chương 2. Nội dung của chương bao gồm việc mô tả bộ dữ liệu, các chỉ số đánh giá, môi trường thực nghiệm, và cuối cùng là trình bày, phân tích sâu các kết quả đã đạt được.

3.1. Bài toán Visual Question Answering

Visual Question Answering là bài toán áp dụng hai kỹ thuật học sâu là xử lý ảnh và xử lý ngôn ngữ tự nhiên. Nhiệm vụ của bài toán này là phân tích một hoặc nhiều hình ảnh và trả lời câu hỏi về hình ảnh tương ứng đó. Bước đầu là phân tích các thông tin đầu vào, bao gồm sử dụng các kỹ thuật xử lý ảnh và xử lý ngôn ngữ tự nhiên để xử lý câu hỏi được đề ra. Sau đó hệ thống VQA phân tích hình ảnh và ngữ cảnh rồi sinh ra câu trả lời phù hợp.



Hình 3.1. Mô hình VQA

Nhiệm vụ hỏi đáp dựa trên hình ảnh cần một mô hình đủ tốt để đưa ra lựa chọn chính xác cho nên việc lựa chọn mô hình rất quan trọng, ngoài ra ta phải tính đến yếu tố về mặt chi phí tính toán, các mô hình ngôn ngữ lớn hiện đại mặc dù đã có xu hướng giảm dần các parameter nhưng vẫn giữ được chức năng tốt so với các mô hình ngôn ngữ lớn lúc mới ra mắt nhưng việc fine-tune mô hình là rất quan trọng đặc biệt là cho các tác vụ cụ thể như VQA.

Bước 1: Thử nghiệm training với model CNN+LSTM truyền thống để đánh giá hiệu năng với bộ dataset

Bước 2: Fine-tune mô hình PaliGemma, so sánh hiệu năng với mô hình CNN+LSTM truyền thống

Bước 3: Đánh giá quá trình fine-tune của mô hình PaliGemma trên thang điểm đánh giá của VQA. dựa theo paper của bộ dataset cách đánh giá sẽ là

$$Acc(ans) = min\left(\frac{\#human\ that\ said\ ans}{3}, 1\right)$$ [14]

Bước 4: Sử dụng các cách optimizer như lượng tử hóa để làm nhẹ mô hình để thực hiện cho quá trình interface trên máy tính cá nhân

3.2. Các công cụ lập trình

3.2.1. Sử dụng ngôn ngữ lập trình python

Là một trong những ngôn ngữ phổ biến bậc nhất trên thế giới đặc biệt cho xử lý dữ liệu cũng như trong lĩnh vực AI

Cú pháp đơn giản, dễ tiếp cận: Khác với nhiều ngôn ngữ lập trình khác, Python nổi bật với cấu trúc rõ ràng, cú pháp đơn giản. Điều này giúp các nhà phát triển AI dễ dàng triển khai ý tưởng, tập trung vào thuật toán và mô hình mà không phải lo lắng quá nhiều về các chi tiết phức tạp của ngôn ngữ

Thư viện và framework phong phú: ngành AI hiện tại không thể tách rời khỏi các thư viện như NumPy, pandas giúp hỗ trợ thao tác ma trận và xử lý dữ liệu, cũng như các framework deep learning như TensorFlow, PyTorch, Keras… Những công cụ này cung cấp sẵn rất nhiều tiện ích từ việc xây dựng mô hình neural network đến huấn luyện, tối ưu và đánh giá kết quả.

Đa dạng ứng dụng trong AI: Từ xử lý bài toán học máy cơ bản như dự đoán, phân loại, clustering… Đến xử lý ngôn ngữ tự nhiên, thị giác máy tính. Python đều đáp ứng tốt. Các thư viện như Spacy, NLTK (về ngôn ngữ) hay OpenCV, scikit-image (về hình

ảnh), hay Hugging Face (về lưu trữ mô hình) chỉ là một vài ví dụ trong vô vàn công cụ hỗ trợ về chuyên môn.

Nhờ những đặc điểm này Python trở thành lựa chọn hàng đầu cho các dự án và nghiên cứu trong lĩnh vực Trí tuệ nhân tạo.

3.2.2. Nền tảng Kaggle

Kaggle là một nền tảng trực tuyến mạnh mẽ, nơi cộng đồng khoa học dữ liệu và học máy cùng phát triển. Ra mắt năm 2010, Kaggle cung cấp môi trường cho các nhà khoa học dữ liệu, nhà phân tích và những người đam mê trí tuệ nhân tạo cùng nhau giải quyết các vấn đề thực tế, chia sẻ kiến thức và tham gia các cuộc thi. Nền tảng này mang đến một kho tài nguyên khổng lồ: các tập dữ liệu, notebook code và hướng dẫn liên quan đến việc học tập và phát triển kỹ năng phân tích đánh giá dữ liệu, phát triển mô hình học máy. Tất cả những yếu tố này, cùng với một cộng đồng mạnh mẽ và bộ công cụ đa dạng, đã giúp Kaggle trở thành một điểm đến toàn diện để phát triển kỹ năng cho hành trình giải quyết các vấn đề phức tạp, dựa trên dữ liệu. Kaggle cung cấp vô số tính năng và dịch vụ đáp ứng mọi cấp độ của những người đam mê khoa học dữ liệu và học máy. Các công cụ và tài nguyên này được thiết kế để người dùng có thể học hỏi, thử nghiệm và cộng tác hiệu quả trong cộng đồng này:

Competitions: Kaggle cung cấp nhiều cuộc thi và thách thức với dữ liệu thực tế từ các doanh nghiệp và tổ chức trên khắp thế giới. Người tham gia có cơ hội đặt câu hỏi, giải quyết các vấn đề thực tế và có cơ hội nhận các giải thưởng hấp dẫn.

Dữ Liệu: Kaggle cung cấp một kho dữ liệu lớn và đa dạng cho cộng đồng. Người dùng có thể tìm thấy các bộ dữ liệu từ nhiều lĩnh vực khác nhau như y học, tài chính, thị trường chứng khoán, thị giác máy tính, và nhiều lĩnh vực khác.

Kernels: Kaggle hỗ trợ sử dụng Jupyter Notebook, giúp người dùng chia sẻ, đọc và chạy mã nguồn một cách dễ dàng. Điều này thúc đẩy sự chia sẻ kiến thức và kinh nghiệm giữa cộng đồng. Ngoài ra Kaggle cung cấp môi trường miễn phí cho người dùng cho việc chạy thử nghiệm các thuật toán và huấn luyện mô hình, Kaggle cung cấp 100GB

lưu trữ dữ liệu, 30 giờ sử dụng GPU trong một tuần, 12 giờ cho một session (P100 15GB , GPU T4x2 15GB).

3.2.3. Thư viện sử dụng trong bài

NumPy: là thư viện cơ bản cho tính toán khoa học trong Python. Nó cung cấp một đối tượng mảng đa chiều, các đối tượng dẫn xuất (như mảng có mặt nạ và ma trận), cùng rất nhiều hàm phục vụ các phép toán nhanh trên mảng, bao gồm toán học, logic, biến đổi hình dạng, sắp xếp, chọn lọc, nhập-xuất, biến đổi Fourier rời rạc, đại số tuyến tính cơ bản, các phép thống kê cơ bản, mô phỏng ngẫu nhiên và nhiều tính năng khác nữa. **torch:** là một framework học sâu mã nguồn mở được thiết kế linh hoạt và có tính module cao, phục vụ cho cả nghiên cứu lẫn triển khai sản phẩm. Nó cung cấp một package Python với các tính năng cấp cao như tính toán tensor (tương tự NumPy) được tăng tốc mạnh mẽ bằng GPU

pandas: là một thư viện mã nguồn mở, cung cấp các cấu trúc dữ liệu hiệu năng cao, dễ sử dụng và các công cụ phân tích dữ liệu cho ngôn ngữ lập trình Python.

Matplotlib: Matplotlib là một thư viện vẽ đồ thị mã nguồn mở mạnh mẽ và linh hoạt dành cho Python, được thiết kế để giúp người dùng trực quan hóa dữ liệu dưới nhiều định dạng khác nhau, thư viện này cho phép người dùng biểu diễn dữ liệu dưới dạng đồ họa, từ đó hỗ trợ việc phân tích và hiểu dữ liệu một cách dễ dàng hơn.

Pillow: là thư viện xử lý ảnh được xây dựng dựa trên PIL (Python 2). Thư viện này hỗ trợ nhiều định dạng tệp hình ảnh như PNG, JPEG, PPM, GIF, TIFF, và BMP.

datasets: là một thư viện cung cấp các công cụ để quản lý và tải dữ liệu từ các nguồn khác nhau, đặc biệt là trong ngữ cảnh machine learning. Thư viện này cung cấp nhiều bộ dữ liệu chuẩn và dễ sử dụng để nghiên cứu và phát triển mô hình.

transformers: Hugging Face Transformers là một thư viện Python mã nguồn mở, cung cấp quyền truy cập vào hàng nghìn mô hình Transformers được huấn luyện sẵn cho các tác vụ xử lý ngôn ngữ tự nhiên (NLP), thị giác máy tính, tác vụ âm thanh và nhiều hơn nữa. Thư viện này đơn giản hóa quy trình triển khai các mô hình Transformer bằng cách

trừu tượng hóa sự phức tạp trong việc huấn luyện hoặc triển khai mô hình trong các framework cấp thấp hơn như PyTorch, TensorFlow và JAX.

peft: (Parameter-Efficient Fine-Tuning) là một thư viện giúp điều chỉnh các mô hình tiền huấn luyện lớn một cách hiệu quả cho nhiều ứng dụng cụ thể mà không cần tinh chỉnh tất cả các tham số của mô hình, bởi vì việc này có chi phí tính toán rất lớn.

bitsandbytes: là thư viện cung cấp các công cụ lượng tử hóa cho các Mô hình Ngôn ngữ Lớn thông qua một trình bao bọc Python gọn nhẹ quanh các hàm CUDA. Thư viện này đặc biệt hỗ trợ các thuật toán tối ưu hóa 8-bit, cùng với các chức năng lượng tử hóa 8-bit và 4-bit.

spaCy: là một thư viện mã nguồn mở phát triển bằng Python, chuyên hỗ trợ các nhiệm vụ xử lý ngôn ngữ tự nhiên như gán nhãn từ loại, nhận diện thực thể có tên, phân tích quan hệ phụ thuộc và nhiều chức năng khác.

Streamlit: là một thư viện Python mã nguồn mở được thiết kế để giúp các developer, data scientist, và machine learning engineer xây dựng các ứng dụng web tương tác mà không cần kỹ năng về front-end.

3.3. Giới thiệu về bộ dữ liệu

Bộ dữ liệu được sử dụng trong đồ án này là Visual Question Answering version 2 (VQA v2.0). Được xây dựng dựa trên bộ dữ liệu MS COCO (Microsoft Common Objects in Context) chứa các hình ảnh (ở dạng RGB và Gray) phức tạp về các cảnh vật đời thường, bộ dữ liệu này được tạo ra nhằm hỗ trợ cộng đồng nghiên cứu phát triển các hệ thống VQA có khả năng suy luận trên các cặp câu hỏi-hình ảnh.

Mỗi cặp câu hỏi-hình ảnh đi kèm với nhiều chú thích (annotations), tức là các câu trả lời khác nhau, đóng vai trò là nhãn thực (ground truth). Các chú thích này sau đó được kết hợp với câu hỏi để làm dữ liệu cho quá trình huấn luyện mô hình. Mặc dù trong một số trường hợp, các câu trả lời chú thích có thể hơi ngắn và chưa đầy đủ, nhưng đây là bộ dữ liệu phù hợp nhất cho mục tiêu nghiên cứu nhờ vào tính cấu trúc và sự thuận tiện trong việc sử dụng.

Split	Number of questions	Number of answers	# of images
Training	443,758	4,437,580	82,783
Validation	214,354	2,143,540	40,504
Testing	447,793	0	81,434

Hình 3.2: Cấu trúc tập dữ liệu VQA v2.0

Bộ dữ liệu được chia thành ba tập con. Với mỗi câu hỏi, có 10 chú thích khác nhau được cung cấp từ những người đánh nhãn khác nhau. Một hình ảnh có thể được liên kết với nhiều câu hỏi khác nhau. Tuy nhiên, bộ dữ liệu vẫn duy trì được sự cân bằng tốt. Trong khuôn khổ của đồ án này, ta chỉ sử dụng 10% đầu tiên của bộ dữ liệu huấn luyện để áp ứng nhu cầu về hiệu năng của phần cứng. Để thuận tiện cho việc truy xuất và xử lý, bộ dữ liệu lưu trữ dưới dạng các tệp JSON gồm hai loại: questions và annotations. Trong mỗi tệp questions bao gồm question_id và image_id và string câu hỏi. Tệp annotations tương ứng chứa cùng uestion_id và image_id kèm theo multiple_choice_answer, answer_type cùng danh sách 10 đáp án gốc do người đánh nhãn cung cấp và một số đáp án plausible được thiết kế để đánh giá khả năng phân biệt trong tình huống khó.

	question_type	multiple_choice_answer	answers	image_id	answer_type	question_id
0	what is this	net	[{'answer': 'net', 'answer_confidence': 'maybe...	458752	other	458752000
1	what	pitcher	[{'answer': 'pitcher', 'answer_confidence': 'y...	458752	other	458752001
2	what color is the	orange	[{'answer': 'orange', 'answer_confidence': 'ye...	458752	other	458752002
3	is this	yes	[{'answer': 'yes', 'answer_confidence': 'yes',...	458752	yes/no	458752003
4	what color is the	white	[{'answer': 'white', 'answer_confidence': 'yes...	262146	other	262146000
...	...	...	...	...	...	...
443752	what color is the	black	[{'answer': 'black', 'answer_confidence': 'yes...	524286	other	524286001
443753	is there a	no	[{'answer': 'no', 'answer_confidence': 'yes', ...	524286	yes/no	524286002
443754	what color is the	black	[{'answer': 'black', 'answer_confidence': 'yes...	524286	other	524286003
443755	why	one is easier to type on	[{'answer': 'don't know', 'answer_confidence':...	524286	other	524286004
443756	is that a	yes	[{'answer': 'yes', 'answer_confidence': 'yes',...	524286	yes/no	524286005

Hình 3.3: Tổng quan tập annotations train

	image_id	question	question_id
0	458752	What is this photo taken looking through?	458752000
1	458752	What position is this man playing?	458752001
2	458752	What color is the players shirt?	458752002
3	458752	Is this man a professional baseball player?	458752003
4	262146	What color is the snow?	262146000
...	...	...	...
443752	524286	What color is the keyboard?	524286001
443753	524286	Is there a computer mouse on the desk?	524286002
443754	524286	What color is the computer?	524286003
443755	524286	Why are there two keyboards?	524286004
443756	524286	Is that a laptop?	524286005

Hình 3.4: Tổng quan tập questions train

```
Sample 1:
Question: What position is this man playing?
Question Type: what
Answer: pitcher
Answer Type: other
```

Hình 3.5: Hình ảnh tương ứng với câu hỏi và trả lời

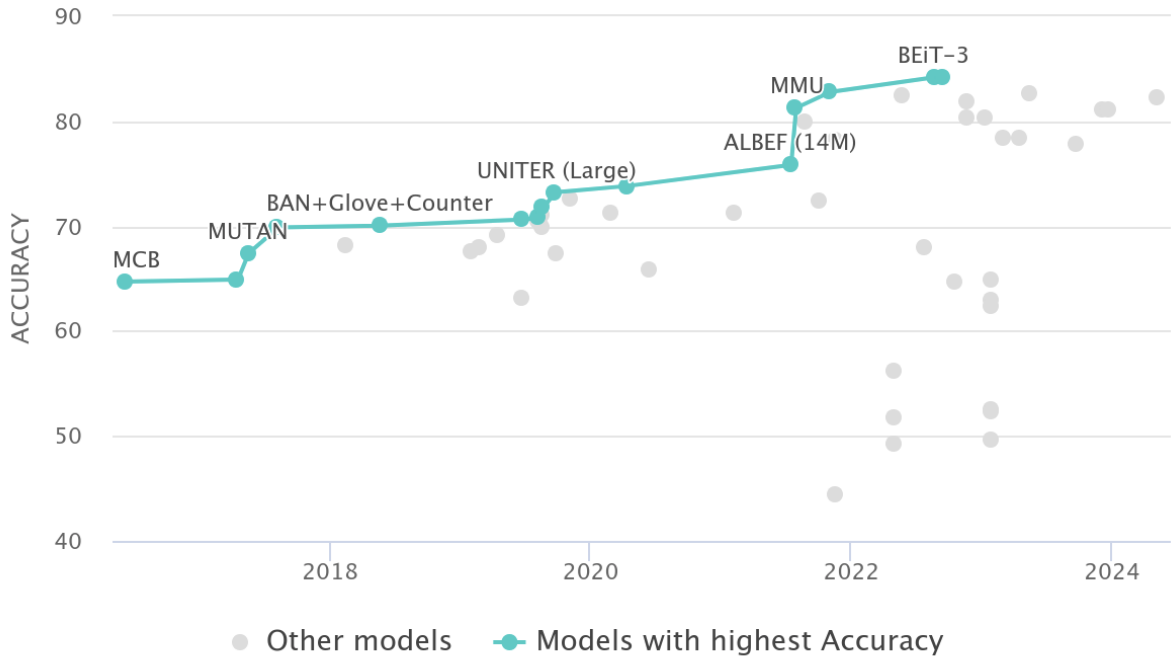
Để kiểm chứng tính đại diện của tập con 10% đầu tiên của bộ dữ liệu huấn luyện, em có thử kiểm tra sự phân bố các loại câu trả lời. Kết quả cho thấy sự phân bố là rất tương đồng: loại câu trả lời “other” chiếm 49.7% trong tập con so với 49.4% trong toàn bộ dữ liệu; loại “yes/no” chiếm 37.1% so với 37.61%; và loại”"number” chiếm 13.2% so với 13.98%. Sự chênh lệch không đáng kể này cho thấy tập dữ liệu được sử dụng trong thực nghiệm vẫn có tính đại diện cao.

	Toàn bộ Dataset (%)	Tập con 10% (%)
answer_type		
other	49.41	49.7
yes/no	37.61	37.1
number	12.98	13.2

Hình 3.6. So sánh độ phân bổ của các loại câu hỏi

CHƯƠNG 4: Thực nghiệm và đánh giá

Việc xây dựng bộ dữ liệu huấn luyện và quy trình cài đặt mô hình cho thuật toán sẽ được trình bày trong chương này. Phần xây dựng chương trình của bài toán sẽ được triển khai bằng Python trên Kaggle và Visual Studio Code. Trước khi vào phần đánh giá ta xem qua bảng xếp hạng các mô hình ngôn ngữ lớn và các mô hình thị giác-ngôn ngữ lớn trên trang web Papers with code (https://paperswithcode.com/sota/visual-question-answering-on-vqa-v2-test-dev) trong khoảng thời gian mà mô hình Paligemma được ra mắt. Ta có thể thấy các mô hình này dù được training, fine-tune trên hàng Terabyte text văn bản cũng như hàng trăm triệu cặp ảnh và caption nhưng vẫn chỉ loanh quanh ở khoảng 78% ~ 84% trên bộ dataset VQAv2.



Hình 4.1: Bảng xếp hạng độ chính xác VQA của các mô hình trong khoảng từ năm 2016 đến 2024 trên trang paperwithcode

Từ biểu đồ trên, chúng ta có thể quan sát thấy một xu hướng tăng trưởng ổn định trong hiệu suất của các mô hình thị giác-ngôn ngữ qua các năm. Bắt đầu từ năm 2018 với mô hình MCB đạt độ chính xác khoảng 65%, các mô hình đã liên tục cải thiện và đạt đến đỉnh cao với BEIT-3 và Pali-3 đạt khoảng 84% vào năm cuối 2022 đầu năm 2023. Việc mô hình chỉ đạt độ chính xác trong khoảng này phản ánh lên sự phức tạp của bài toán

VQA. Bài toán buộc các mô hình phải thực hiện khả năng suy luận phức tạp, đa bước, phải làm tốt cả về mặt xử lý ảnh lẫn xử lý ngôn ngữ tự nhiên.

4.1. Training mô hình CNN+LSTM

4.1.1. Tiền xử lý dữ liệu

Tiền xử lý dữ liệu là bước khởi đầu và đóng vai trò then chốt trong việc giải quyết bất kỳ bài toán nào thuộc lĩnh vực trí tuệ nhân tạo. Để mô hình đạt được độ chính xác cao, việc xử lý và chuẩn bị dữ liệu một cách cẩn thận là yếu tố không thể thiếu đối với mọi hệ thống học máy. Trước khi vào quá trình training, ta tiến hành các bước như sau:

1. Loại bỏ các cột không cần thiết: Ta loại bỏ đi các thuộc tính không cần thiết trong quá trình training là question_type, answers, answer_type, image_id, và question_id chỉ giữ lại các cột như image, question, và multiple_choice_answer
2. Chuẩn hóa các câu trả lời về lowercase, loại bỏ khoảng trắng, loại bỏ dấu câu và chuyển số thành từ
3. Vì VQA dataset có thể chứa hàng chục nghìn câu trả lời khác nhau, ta chỉ lựa chọn 1000 câu trả lời phổ biến nhất để tạo thành không gian phân loại.
4. Hình ảnh được resize về kích thước 224x224 pixels và được chuẩn hóa theo ImageNet statistics. Đối với tập training, các kỹ thuật data augmentation như random crop, color jitter và horizontal flip được áp dụng để tăng tính đa dạng của model. Tập validation chỉ được resize và normalize để đảm bảo tính nhất quán trong đánh giá.
5. Những samples có câu trả lời không nằm trong top 1000 được gán label = -1 và bị ignore trong quá trình training thông qua ignore_index parameter trong loss function.

4.1.2. Kiến trúc mô hình

Để thiết lập mô hình so sánh cho bài toán, em sử dụng một kiến trúc kết hợp của mạng CNN và LSTM có khả năng xử lý đồng thời hai luồng dữ liệu hình ảnh và văn bản, sau đó kết hợp lại để đưa ra dự đoán. Kiến trúc của mô hình gồm:

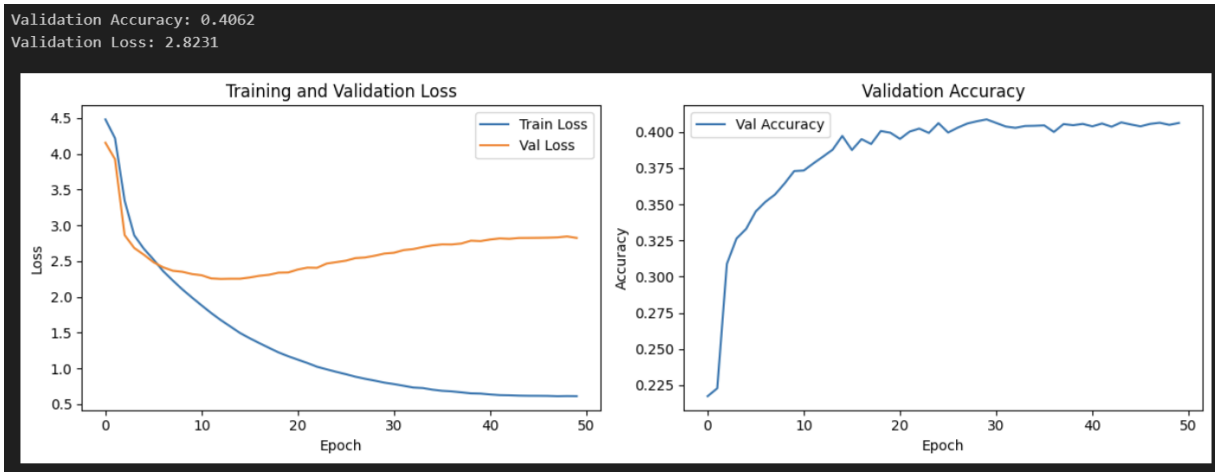
- Bộ mã hóa ảnh: Sử dụng kiến trúc mạng CNN là ResNET-50 đã được pre-train từ bộ dữ liệu ImageNET. Ảnh đầu vào đã được xử lý là ảnh màu với kích thước 224x224 cùng kích thước với PaliGemma.
- Bộ mã hóa câu hỏi: Sử dụng mạng LTSM hai chiều (Bidirectional LSTM) gồm 2-layer xếp chồng lên nhau. Có sử dụng dropout là 30% giữa các lớp LSTM để giảm thiểu overfitting
- Khối kết hợp: Sử dụng cơ chế chiếu (projection) để vector đầu ra của cả ảnh và câu hỏi điều về cùng chiều sau đó concatenate lại với nhau. Sau đó vector được nối này sẽ đi qua một lớp Linear để giảm chiều xuống. Hàm kích hoạt được sử dụng là GeLU. Lớp dropout là 30%.

Các tham số huấn luyện được sử dụng trong mô hình là:

- Hàm mất mát là CrossEntropy
- Optimizer là AdamW (weight decay là 1e-4)
- Learning rate là 1e-4
- Số epoch là 50
- Batch size là 128

4.1.3. Kết quả huấn luyện

Sử dụng 10% của tập data VQA trên Huggingface, chia 90-10 cho tập train và test. Mô hình CNN ở đây là ResNET50 và Bi-LSTM. Tiếp theo là chạy code training mô hình trên 50 epoch và hoàn thành sau hơn 7 tiếng



Hình 4.2. Kết quả training của mô hình CNN+LSTM

Mặc dù đã áp dụng một loạt các kỹ thuật như data Augmentation, áp dụng dropout, sử dụng L2 regularization là weight decay nhưng với bộ dataset đòi hỏi mức suy luận cao như VQA thì mô hình đã gặp phải trường hợp overfitting mạnh. Cũng như độ chính xác

của mô hình chỉ đạt xấp xỉ 40%. Từ đây ta có thể thấy mô hình truyền thống như CNN+LSTM gặp phải rào cản cực kì lớn với bài toán này.

4.2. Fine-tune mô hình PaliGemma

4.2.1. Tiền xử lý dữ liệu

Các bước tiền xử đầu của PaliGemma cũng giống như của mô hình CNN+LSTM. Đầu tiên cũng loại bỏ các cột không cần khi training, sau đó sẽ khác so với mô hình trên:

1. Mỗi câu hỏi sẽ được thêm tiền tố “answer”. Điều này tận dụng khả năng hiểu ngữ cảnh tốt của PaliGemma
2. Chuyển hình ảnh sang dạng RGB và được tokenize cùng với text thành một chuỗi token duy nhất
3. Chuyển toàn bộ định dạng từ float32 sang float16 để tăng tốc độ tính toán cho mô hình
4. Sử dụng "longest padding" để tối ưu batch processing. Processor tự động resize và normalize hình ảnh theo chuẩn mô hình mà không cần data augmentation như CNN+LSTM

4.2.2. Fine-tune mô hình

Đầu tiên ta tinh chỉnh các tham số của mô hình, trong đó vẫn sử dụng 10% data của VQA trên Huggingface và chia 90-10, sau đó tiến hành fine-tune:

```
args=TrainingArguments(
     num_train_epochs=5, “đã thử với 1,2 và 10 ”
     emove_unused_columns=False,
     per_device_train_batch_size=4,
     gradient_accumulation_steps=4,
     warmup_steps=5,
     learning_rate=2e-5,”đã thử với 1e-4 và 1e-5”
     weight_decay=1e-6,
     adam_beta2=0.999,
     logging_steps=100,
     save_strategy="steps",
     save_steps=1000,
     push_to_hub=True,
     save_total_limit=1,
     output_dir="paligemma_vqav2_10pc",
     bf16=True,
     report_to=["tensorboard"],
     dataloader_pin_memory=False
)
trainer=Trainer(
     model=model,
     train_dataset=train_ds,
     eval_dataset=val_ds,
     data_collator=collate_fn,
     args=args
) [15]
```

Giải thích tham số:

num_train_epochs: Số lần duyệt qua toàn bộ dữ liệu huấn luyện

emove_unused_columns: Giữ nguyên toàn bộ tất cả các cột trong bộ dữ liệu

per_device_train_batch_size: kích thước batch cho mỗi GPU

gradient_accumulation_steps: Số bước tích lũy gradient trước khi cập nhập trọng số

warmup_steps: số bước mà tốc độ học tăng dần từ số rất nhỏ đến tốc độ học mong muốn

learning_rate: tốc độ học của mô hình

weight_decay: hệ số L2 Regularization

adam_beta2: tham số của optimizer Adam, điều khiển momentum của gradient bậc 2

logging_steps: ghi lại log mỗi sau mỗi bước training lên Huggingface

report_to: gửi các metric đánh giá lên Huggingface

save_strategy: cách thức lưu model

save_steps: lưu checkpoint sau mỗi bước

save_total_limit: giữ lại checkpoint sau các lần gần nhất

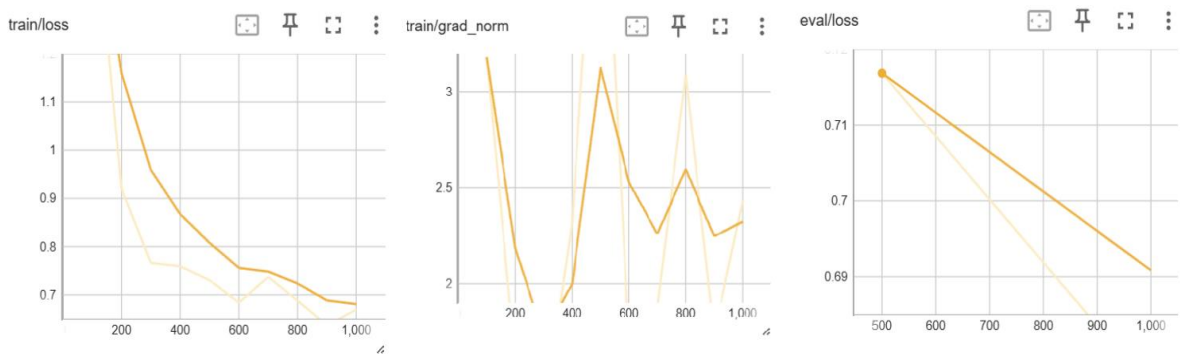
output_dir: thư mục lưu checkpoint

push_to_hub: Lựa chọn có đẩy lên Huggingface không

bf16: sử dụng precision bfloat16

dataloader_pin_memory: lựa chọn có pin lại memory cho dataloader không

Chạy code fine-tune mô hình trên Kaggle (GPU T4x2) với 5 epoch sau 12 tiếng kết quả đạt được như sau:



Hình 4.3. Kết quả fine-tune của PaliGemma

Qua nhiều lần chỉnh sửa các hyperparameter và fine-tune em đã đạt được kết quả train loss chỉ 0.6688, giá trị validation loss là 0.6752 và giá trị gradient norm chỉ nằm trong khoảng từ 2 cho đến 3 không quá cao nhưng cũng không thấp chứng tỏ quá trình training khá ổn định và ít xảy ra trường hợp overfitting. Nhưng để đánh giá kĩ hơn mô hình ta phải đánh giá theo chuẩn của bài toán VQA. Toàn bộ kết quả sau các lần training em để tại https://huggingface.co/gintorikj/paligemma_vqav2_10pc

4.3. Đánh giá mô hình PaliGemma

Trước khi đi vào hàm đánh giá ta vẫn phải xử lý data đầu vào, chuẩn hóa về chữ thường, loại bỏ các từ không cần thiết như a, an, the, chuyển ảnh về dạng RGB, xóa các kí tự đặc biệt, khoảng trắng đi sau đó ta mới xây dựng hàm tính độ chính xác và đánh giá. Sau đây là các kết quả sau khi đánh giá qua 5000 sample ảnh trên bộ data (đã thử qua với 2000 - 8000 - 10000 sample. Kết quả khi dự đoán xấp xỉ bằng nhau và load quá 10000 ảnh sẽ có hiện tượng bị treo session khi lượng ram đạt đến gần 30GB). Đầu tiền ta xây dựng hàm đánh giá:

```
def vqa_accuracy(predicted_answers, ground_truth_answers_list):
     detailed_scores = []
     for pred, gt_answers in zip(predicted_answers,
                                 ground_truth_answers_list):
          pred_normalized = normalize_answer(pred)
          answer_counts = Counter(normalize_answer(gt)
                                 for gt in gt_answers)
          score = 0.0
          if pred_normalized in answer_counts:
               score=min(answer_counts[pred_normalized]/3.0,1.)
          detailed_scores.append(score)
     if detailed_scores:
          accuracy = sum(detailed_scores)/len(detailed_scores)
     else:
          accuracy = 0.0
     return accuracy, detailed_scores
```

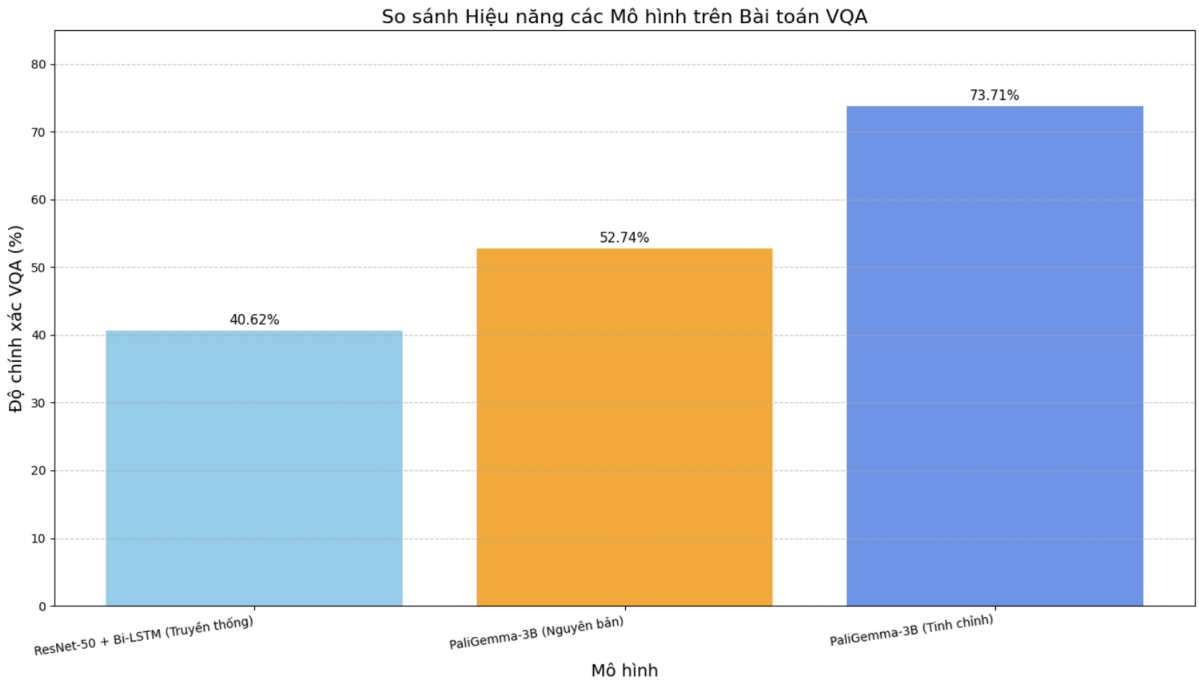
Ta sẽ đánh giá mô hình base của PaliGemma sau đó là PaliGemma sau khi fine-tune:

```
KẾT QUẢ ĐÁNH GIÁ VQA
Model:google/paligemma-3b-pt-224
Dataset: VQAv2
Số mẫu được đánh giá: 5000
VQA Accuracy: 52.74%
```

Hình 4.4. Kết quả đánh giá mô hình PaliGemma nguyên bản

```
KẾT QUẢ ĐÁNH GIÁ VQA
Model: gintorikj/paligemma_vqav2_10pc
Dataset: VQAv2
Số mẫu được đánh giá: 5000
VQA Accuracy: 73.71%
```

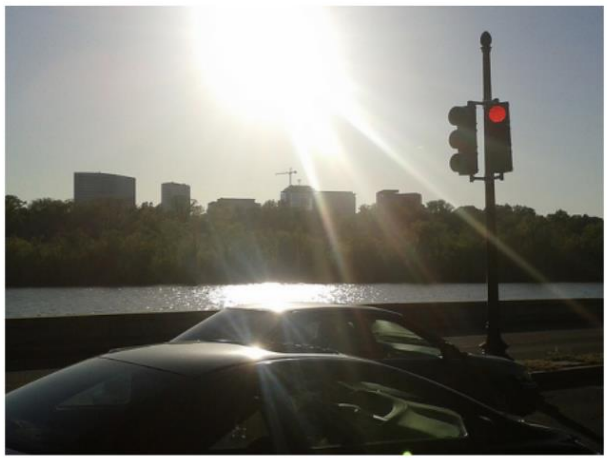
Hình 4.5. Kết quả đánh giá mô hình PaliGemma sau khi tinh chỉnh



Hình 4.6. So sánh độ chính xác VQA của các mô hình

Sau khi chạy 5000 sample, kết quả cho thấy mô hình đạt được kết quả khá tốt với chỉ 10% dữ liệu huấn luyện VQA, chứng tỏ khả năng học nhanh và tổng quát hóa tốt của PaliGemma rất tốt với lượng dữ liệu hạn chế. tốt hơn rất nhiều so với mô hình CNN+LSTM truyền thống và xem lại bảng xếp hạng độ chính xác VQA ở *Hình 4.1* nó tốt khá nhiều so với các mô hình cùng thời dù chỉ train trên 10% data.

Câu hỏi: How lucky is it that no one was walking on the sidewalk at that time?



```
Model dự đoán: very
Đáp án gốc (Ground Truth): ['no danger', 'very lucky', 'somewhat', 'lucky', 'very', 'very', 'not', 'lucky', 'very', 'very']
Điểm VQA cho câu này: 1.00
```

Hình 4.7. Mô hình dự đoán đúng

Câu hỏi: What is the figure on the cupcake?

```
Model dự đoán: surfer
Đáp án gốc (Ground Truth): ['soldier', 'human', 'soldier', 'soldier', 'army', 'army man', 'soldier', 'army man', 'soldier', 'soldier']
Điểm VQA cho câu này: 0.00
```

Hình 4.8. Mô hình dự đoán sai

Kết quả thực nghiệm không chỉ được đo lường qua các chỉ số định lượng mà còn được đánh giá sâu sắc thông qua quá trình phân tích định tính. Quá trình này cho thấy mô hình đề xuất có khả năng trả lời chính xác nhiều câu hỏi có độ khó cao, vốn đòi hỏi khả năng suy luận phức tạp ngay cả với con người. Và trong quá trình visualize loạt các hình ảnh dự đoán của mô hình, có rất nhiều hình ảnh mà kể cả con người cũng khó có thể xác định đúng xuất hiện. Có những câu hỏi có tới 6 đến 10 đáp án khác nhau được cung cấp bởi người đánh nhãn. Điều này càng làm nhấn mạnh thêm độ khó của bài toán khi mô hình phải đối mặt với các câu hỏi có tính mở và phải hiểu ngữ cảnh.

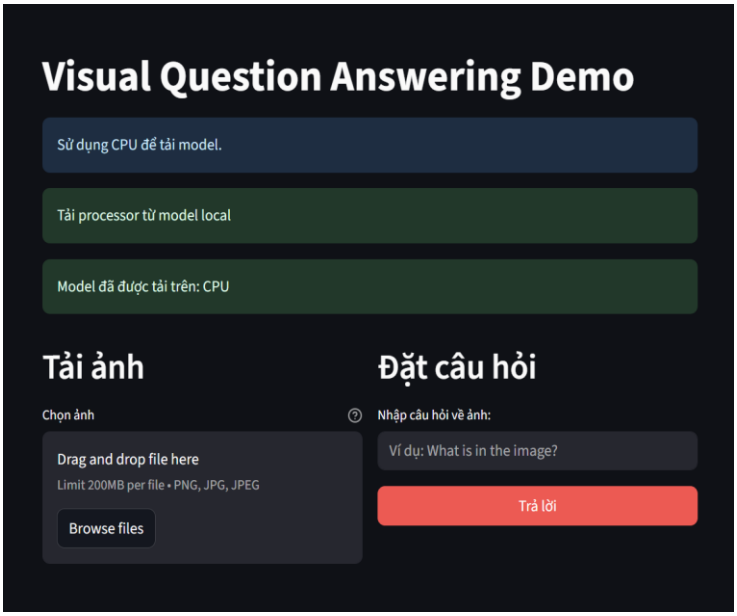
Hình 4.9. Câu hỏi thiếu thông tin, không rõ ràng dẫn đến nhiều đáp án khác nhau

4.4. Xây dựng giao diện demo

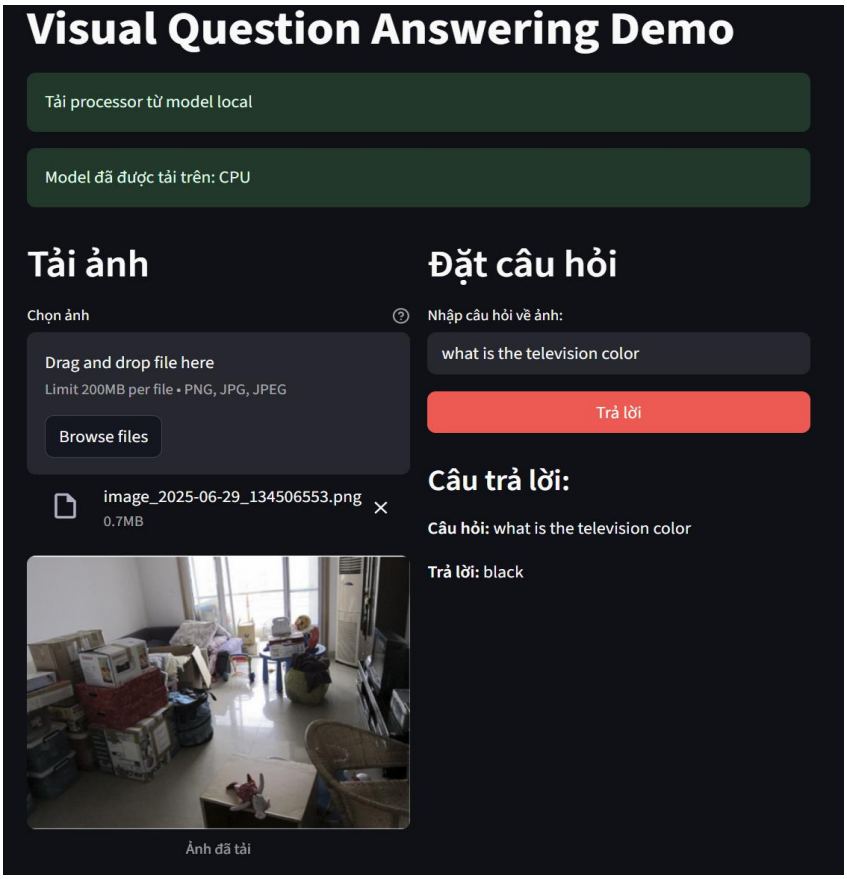
Tổng quan: chương trình VQA được tạo bởi streamlit có chứng năng chính gồm chọn ảnh (có 3 định dạng PNG, JPG, JPEG) từ máy tính giới hạn 200MB, nhập câu hỏi của mình bằng tiếng anh cho mô hình. Mô hình sẽ lấy trọng số được lưu sẵn ở trong máy, load trên CPU và trả lời lại câu hỏi mà ta vừa nhập.

Quá trình vận hành thử nghiệm giao diện được tiến hành trên máy tính cá nhân có cấu hình CPU AMD Ryzen 7 5800HS và 24GB RAM. Một quan sát quan trọng trong quá trình này là mức tiêu thụ tài nguyên hệ thống rất lớn. Cụ thể, khi tải và vận hành mô hình trên CPU, chương trình yêu cầu tới 22GB RAM, chiếm gần như toàn bộ dung lượng

bộ nhớ khả dụng. Mức tiêu thụ bộ nhớ cao này chủ yếu xuất phát từ kích thước vốn có của các mô hình ngôn ngữ-thị giác lớn như PaliGemma dù đã được lượng tử hóa.



Hình 4.10. Giao diện của chương trình



Hình 4.11. Chạy thử chương trình

CHƯƠNG 5: Kết luận và phương hướng phát triển

Sau quá trình nghiên cứu, triển khai thực nghiệm và đánh giá, đồ án **"Xây dựng mô hình học sâu đa phương thức cho nhiệm vụ hỏi đáp dựa trên hình ảnh"** đã hoàn thành các mục tiêu cốt lõi đã đặt ra. Chương này sẽ hệ thống hóa những kết quả đạt được, chỉ rõ các đóng góp của đề tài, phân tích các hạn chế còn tồn tại và đề xuất những hướng phát triển tiềm năng trong tương lai.

5.1. Kết luận

Đề tài đã tiếp cận bài toán Hỏi đáp dựa trên hình ảnh đã và đang được cộng đồng học máy quan tâm trong thời gian gần đây. Mục tiêu của các nghiên cứu nhằm tìm ra phương pháp xây dựng mô hình có khả năng suy nghĩ, hiểu ngữ cảnh, hoạt động chính xác và hiệu quả. Về mặt kết quả thực nghiệm, mô hình PaliGemma đã có thấy sự vượt trội rệt so với các mô hình CNN+LSTM truyền thống. Quá trình này không chỉ giúp đánh giá năng lực của từng mô hình mà còn phản ánh sự dịch chuyển công nghệ trong lĩnh vực Trí tuệ Nhân tạo đa phương thức.

Ngoài ra, việc triển khai lượng tử hóa cũng giúp các mô hình ngôn ngữ lớn và mô hình thị giác-ngôn ngữ lớn có thể chạy trên phần cứng hạn chế của máy tính. Qua thực nghiệm, dù chỉ chạy trên CPU nhưng một mô hình có 3 tỷ tham số được lượng tử hóa phản hồi và cho ra kết quả khá nhanh. Tuy nhiên, vẫn tồn tại rất nhiều hạn chế mà bài toán cần giải quyết

5.2. Hạn chế

Bên cạnh những kết quả tích cực, đề tài cũng nhìn nhận những hạn chế khách sau:

- Giới hạn về mặt dữ liệu: Do giới hạn về tài nguyên tính toán, toàn bộ quá trình thực nghiệm chỉ được tiến hành trên 10% bộ dữ liệu VQAv2. Mặc dù hiệu năng của mô hình rất tốt với lượng data ít ỏi nhưng vẫn chưa phản ánh được hết hiệu năng của mô hình. Kết quả là dù có phần trăm độ chính xác cao hơn so với các mô hình cùng thời nhưng chưa thể so sánh một cách khách quan nhất

- Chưa tối ưu được trên GPU: Giao diện chỉ demo trên dược CPU vì vậy, chưa thể đánh giá hết được hiệu năng thật sự của mô hình.
- Chưa trả lời tốt các hỏi cần hiểu ngữ cảnh cao và những câu hỏi dạng number.

5.3. Hướng phát triển trong tương lai

Đầu tiên là khắc phục được các các hạn chế kể trên. Sau đó, các hướng phát triển không chỉ dừng lại ở việc cải thiện các chỉ số mà còn mở ra những hướng nghiên và ứng dụng mới:

- Thay vì chỉ nhìn vào độ chính xác tổng thể, nghiên cứu trong tương lai cần phân tích và cải thiện các dạng suy luận cụ thể mà mô hình còn yếu. Ví dụ: suy luận về số lượng, không gian, …
- Phát triển các ứng dụng thực tiễn: hướng tới các mô hình có thể giải thích được. Thay vì chỉ đưa ra câu trả lời, mô hình cần có khả năng chỉ ra vùng ảnh và những từ khóa trong câu hỏi mà nó đã "chú ý" đến để đưa ra quyết định. Việc này giúp tăng độ tin cậy và cho phép con người hiểu được "suy nghĩ" của mô hình.
- Tích hợp được Tiếng Việt vào mô hình: Công việc này không chỉ đơn thuần là dịch thuật mà còn đòi hỏi việc xây dựng các bộ dữ liệu mới, phản ánh đúng các đặc thù về văn hóa, xã hội và ngôn ngữ của Việt Nam để mô hình có thể thực sự hữu ích trong bối cảnh trong nước.

TÀI LIỆU THAM KHẢO

[1] Cem Akkus, Luyang Chu, Vladana Djakovic, et al, "Multimodal Deep Learning" arXiv:2301.04856 , 2023.

[2] Lucas Beyer, Andreas Steiner, André Susano Pinto, et al, "PaliGemma: A versatile 3B VLM for transfer", arXiv:2407.07726, 2024.

[3] Alec Radford, Jong Wook Kim, Chris Hallacy, et al, "Learning Transferable Visual Models From Natural Language Supervision", arXiv:2103.00020, 2021.

[4] Xiaohua Zhai, Basil Mustafa, Alexander Kolesnikov, et al, "Sigmoid Loss for Language Image Pre-Training", arXiv:2303.15343, 2023.

[5] A. Dosovitskiy, L. Beyer, A. Kolesnikov, et al, "An Image is Worth 16x16 Words: Transformers for Image Recognition at Scale", arXiv:2010.11929, 2021.

[6] Ashish Vaswani, Noam Shazeer, Niki Parmar, et al, "Attention Is All You Need", arXiv:1706.03762, 2017.

[7] Kaiming He, Xiangyu Zhang, Shaoqing Ren, et al, "Deep Residual Learning for Image Recognition", arXiv:1512.03385, 2015.

[8] Jimmy Lei Ba, Jamie Ryan Kiros, Geoffrey E. Hinton, "Layer Normalization", arXiv:1607.06450, 2016.

[9] Noam Shazeer, "Fast Transformer Decoding: One Write-Head is All You Need", arXiv:1911.02150, 2019.

[10] Jianlin Su, Yu Lu, Shengfeng Pan, et al, "RoFormer: Enhanced Transformer with Rotary Position Embedding", arXiv:2104.09864, 2021.

[11] Noam Shazeer, "GLU Variants Improve Transformer", arXiv:2002.05202, 2020.

[12] Dan Hendrycks, Kevin Gimpel, "Gaussian Error Linear Units (GELUs)", arXiv:1606.08415, 2023.

[13] Biao Zhang, Rico Sennrich, "Root Mean Square Layer Normalization", arXiv:1910.07467, 2019.

[14] Aishwarya Agrawal, Jiasen Lu, Stanislaw Antol, et al, "VQA: Visual Question Answering", arXiv:1505.00468, 2015.

[15] merve, Andreas P. Steiner, Pedro Cuenca, "PaliGemma – Google's Cutting-Edge Open Vision Language Model", 2024, [Online]. Available: https://huggingface.co/blog/paligemma