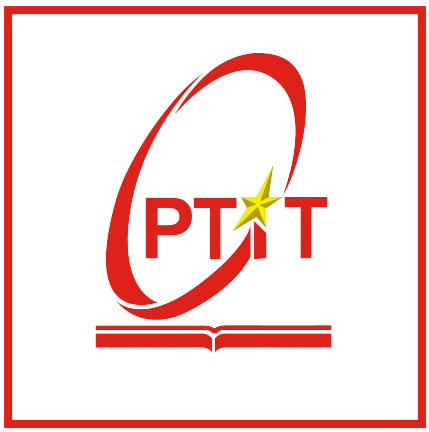
HỌC VIỆN CÔNG NGHỆ BƯU CHÍNH VIỄN THÔNG
Khoa Công nghệ Thông tin 1



BÀI TẬP LỚN THỰC TẬP CƠ SỞ

Đề tài: XÂY DỰNG MÔ HÌNH TRẢ LỜI CÂU HỎI TỪ HÌNH ẢNH

Lớp: TTCS 33
Số thứ tự nhóm: 03

Trần Gia Hiển MSSV: B22DCCN291
Đặng Ngọc Bách MSSV: B22DCCN051
Lò Văn Quyền MSSV: B22DCCN674

Giảng viên hướng dẫn: Ths. Vũ Hoài Thư

Hà Nội, 04/2025

Lời cảm ơn

Nhóm nghiên cứu xin gửi lời tri ân sâu sắc đến Học viện Công nghệ Bưu chính Viễn thông, đặc biệt là Khoa Công nghệ Thông tin 1, đã cung cấp môi trường học tập lý tưởng với cơ sở vật chất hiện đại, tài liệu phong phú và điều kiện thuận lợi để thực hiện bài tập lớn. Sự hỗ trợ của nhà trường giúp nhóm nghiên cứu tiếp cận công nghệ tiên tiến, nâng cao kiến thức và kỹ năng trong lĩnh vực trí tuệ nhân tạo. Đặc biệt cảm ơn cô Vũ Hoài Thư, giảng viên hướng dẫn, đã tận tâm dẫn dắt, khơi dậy đam mê và giúp nhóm nghiên cứu trưởng thành trong tư duy khoa học. Cảm ơn bạn bè đã đồng hành, chia sẻ ý tưởng và động viên. Dù đã nỗ lực, đề tài vẫn còn thiếu sót. Nhóm nghiên cứu mong nhận ý kiến đóng góp để hoàn thiện hơn. Xin tri ân tất cả vì sự hỗ trợ quý báu.

Hà Nội, tháng 04 năm 2025

Mục lục

DANH MỤC HÌNH ẢNH

DANH MỤC BẢNG BIỂU

DANH MỤC THUẬT NGỮ VÀ TỪ VIẾT TẮT

Tên	**Viết đầy đủ tiếng Anh**	**Giải thích ngắn gọn**
AI	Artificial Intelligence	Máy móc có khả năng tư duy
VQA	Vision Question Answering	Trả lời câu hỏi từ hình ảnh
NLP	Natural Language Processing	Xử lý và hiểu ngôn ngữ tự nhiên
CNN	Convolutional Neural Network	Mạng học sâu xử lý hình ảnh
LSTM	Long Short-Term Memory	Mạng học sâu xử lý chuỗi dữ liệu
BERT	Bidirectional Encoder Representations from Transformers	Mô hình ngôn ngữ học ngữ cảnh
KD	Knowledge Distillation	Kỹ thuật nén mô hình học sâu
ReLU	Rectified Linear Unit	Hàm kích hoạt trong mạng nơron
MLM	Masked Language Model	Mô hình dự đoán từ bị che
NSP	Next Sentence Prediction	Dự đoán mối quan hệ giữa câu
Bi-LSTM	Bidirectional LSTM	Mạng LSTM xử lý hai chiều
ResNet50	Residual Network with 50 Layers	Mạng học sâu với 50 lớp
Adam	Adaptive Moment Estimation	Phương pháp tối ưu hóa học sâu
LambdaLR	Lambda Learning Rate	Lịch trình điều chỉnh tốc độ học

1 Giới thiệu đề tài

1.1 Đặt vấn đề

Trong bối cảnh cuộc cách mạng công nghiệp 4.0, trí tuệ nhân tạo (AI) đang trở thành động lực thúc đẩy sự đổi mới trong nhiều lĩnh vực, từ chăm sóc sức khỏe, giáo dục đến giải trí. Một trong những hướng nghiên cứu nổi bật của AI là Vision Question Answering (VQA), kết hợp xử lý ngôn ngữ tự nhiên (NLP) và thị giác máy tính (Computer Vision) để tạo ra các hệ thống có khả năng hiểu và trả lời câu hỏi dựa trên nội dung hình ảnh. VQA không chỉ mở ra tiềm năng cho các ứng dụng thực tiễn như trợ lý ảo thông minh, công cụ tìm kiếm hình ảnh, hay hỗ trợ người khiếm thị, mà còn đặt ra những thách thức lớn về mặt kỹ thuật trong việc tích hợp và tối ưu hóa các mô hình học sâu.

Xuất phát từ tầm quan trọng của VQA và nhu cầu ứng dụng thực tiễn, đề tài "Vision Question Answering" được thực hiện nhằm khám phá và phát triển một hệ thống VQA cơ bản, tập trung vào trả lời các câu hỏi dạng Yes/No dựa trên hình ảnh, đồng thời tận dụng các kỹ thuật học sâu tiên tiến để đảm bảo hiệu quả và tính khả thi.

1.2 Mục tiêu và phạm vi đề tài

Mục tiêu:

- Xây dựng một hệ thống VQA đơn giản có khả năng trả lời câu hỏi Yes/No dựa trên hình ảnh, sử dụng các mô hình học sâu như CNN và LSTM.
- Nâng cao hiểu biết về các phương pháp xử lý hình ảnh và văn bản trong AI, đồng thời rèn luyện kỹ năng áp dụng các kỹ thuật học sâu vào bài toán thực tiễn.
- Đặt nền móng cho các nghiên cứu sâu hơn về VQA và góp phần vào sự phát triển của lĩnh vực trí tuệ nhân tạo tại Việt Nam.

Phạm vi:

- Đề tài tập trung vào việc xử lý câu hỏi Yes/No dựa trên tập dữ liệu VQA v2.0, sử dụng BERT Tokenizer cho tiền xử lý văn bản và phương pháp Knowledge Distillation để tối ưu hóa mô hình.
- Các bước thực hiện bao gồm thu thập, tiền xử lý dữ liệu, xây dựng và huấn luyện mô hình, đánh giá hiệu suất, và áp dụng cho dữ liệu thực tế.

1.3 Định hướng giải pháp

Để giải quyết bài toán VQA, nhóm đề xuất sử dụng mô hình học sâu kết hợp:

- **Xử lý hình ảnh:** Sử dụng mô hình CNN tùy chỉnh để trích xuất đặc trưng hình ảnh.
- **Xử lý văn bản:** Áp dụng BERT Tokenizer để tiền xử lý văn bản và LSTM để trích xuất đặc trưng câu hỏi.
- **Tối ưu hóa mô hình:** Sử dụng phương pháp Knowledge Distillation hướng đến việc giảm độ phức tạp tính toán mà vẫn đảm bảo hiệu suất của mô hình.

- **Huấn luyện và đánh giá:** Tối ưu hóa mô hình bằng optimizer Adam, scheduler LambdaLR , và đánh giá hiệu suất trên các tập dữ liệu validation/test để đảm bảo khả năng tổng quát hóa.

1.4 Bố cục bài tập lớn

Phần còn lại của báo cáo được tổ chức như sau:

Chương 2 giới thiệu lý thuyết VQA, gồm CNN, LSTM, BERT Tokenizer, và Knowledge Distillation, dựa trên tài liệu khoa học uy tín.

Chương 3 trình bày thực nghiệm và đánh giá mô hình VQA Yes/No trên VQA v2.0, từ thiết lập, huấn luyện, đến triển khai trên Streamlit.

Chương 4 tóm tắt kết quả, đề xuất cải thiện độ chính xác và ứng dụng thực tiễn cho VQA tại Việt Nam.

2 Cơ sở lý thuyết

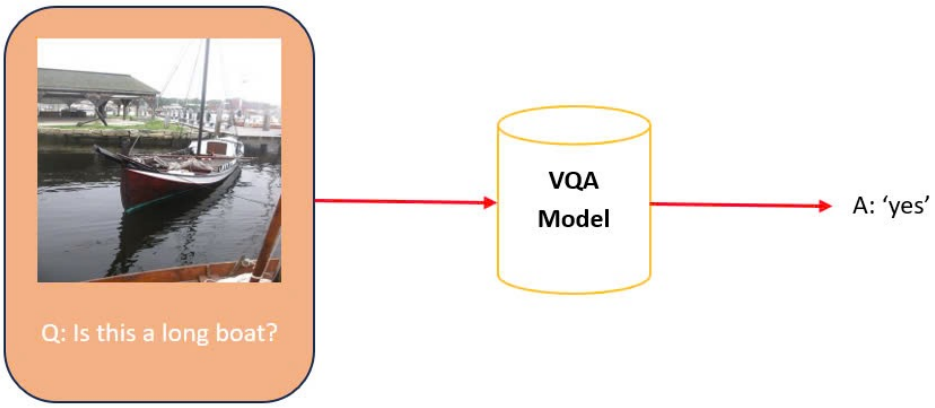
Chương này trình bày các kiến thức nền tảng và cơ sở lý thuyết liên quan đến bài toán Vision Question Answering (VQA), tập trung vào các công nghệ và thuật toán được sử dụng trong đề tài, bao gồm mạng nơ-ron tích chập (CNN), mạng nơ-ron hồi tiếp dài ngắn hạn (LSTM), BERT Tokenizer, và phương pháp Knowledge Distillation (KD). Các nội dung được tóm tắt từ các tài liệu khoa học uy tín, đảm bảo tính liên kết với phần giới thiệu đề tài và các phương pháp triển khai thực tế trong báo cáo.

2.1 Giới thiệu về bài toán Vision Question Answering

Vision Question Answering (VQA) là một bài toán kết hợp xử lý ngôn ngữ tự nhiên (NLP) và thị giác máy tính (Computer Vision), nhằm xây dựng hệ thống có khả năng trả lời các câu hỏi dựa trên nội dung của hình ảnh. Bài toán yêu cầu mô hình hiểu được cả thông tin hình ảnh (ví dụ, đối tượng, màu sắc, vị trí) và ngữ nghĩa của câu hỏi (ví dụ, cấu trúc ngữ pháp, ý định), sau đó đưa ra câu trả lời chính xác, thường ở dạng Yes/No hoặc văn bản ngắn. VQA có ứng dụng rộng rãi, từ trợ lý ảo thông minh, công cụ tìm kiếm hình ảnh, đến hỗ trợ người khiếm thị.

Bài toán VQA đặt ra thách thức trong việc tích hợp hai luồng dữ liệu: hình ảnh và văn bản. Thông thường, một hệ thống VQA bao gồm:

- Một mô hình xử lý hình ảnh để trích xuất đặc trưng (feature extraction) từ hình ảnh.
- Một mô hình xử lý văn bản để mã hóa và hiểu câu hỏi.
- Một cơ chế kết hợp (fusion) để tích hợp đặc trưng hình ảnh và văn bản, sau đó đưa ra dự đoán.

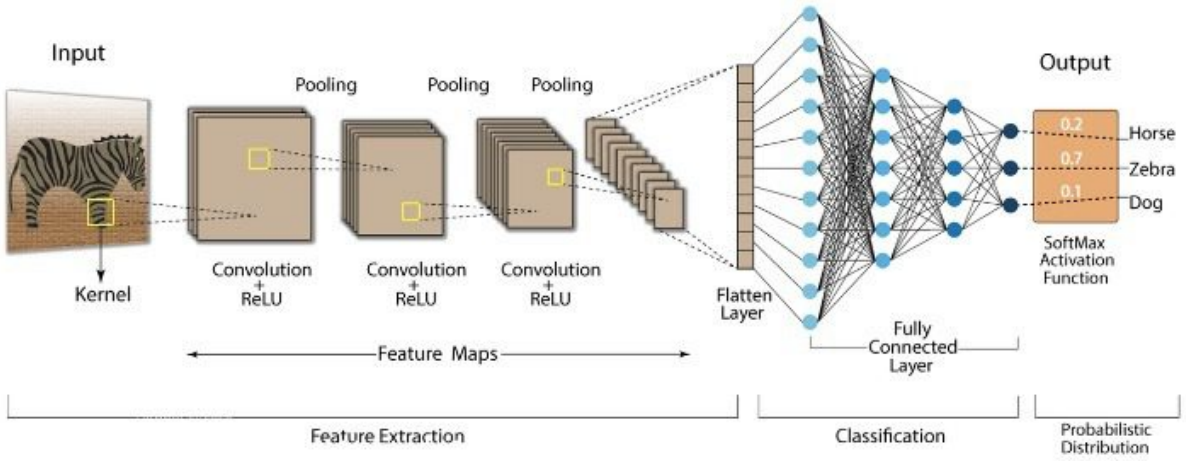


Hình 2.1: Minh hoạ cho VQA

Trong đề tài này, nhóm nghiên cứu tập trung vào câu hỏi dạng Yes/No, sử dụng tập dữ liệu VQA v2.0, với các công nghệ chính là CNN, LSTM.

2.2 Mạng nơ-ron tích chập (CNN)

Mạng nơ-ron tích chập (Convolutional Neural Networks - CNN) là một lớp mô hình học sâu được thiết kế để xử lý dữ liệu có cấu trúc lưới, như hình ảnh hoặc chuỗi thời gian [4]. CNN sử dụng các lớp tích chập (convolutional layers) để trích xuất đặc trưng không gian, chẳng hạn như cạnh, góc, hoặc các mẫu phức tạp hơn, từ dữ liệu đầu vào. CNN đặc biệt hiệu quả trong các bài toán nhận diện hình ảnh, phân loại đối tượng, và xử lý ngôn ngữ tự nhiên.



Hình 2.2: Kiến trúc tổng quan của CNN[20]

2.2.1 Kiến trúc của CNN

Kiến trúc CNN điển hình bao gồm các lớp chính sau:

Lớp tích chập (Convolutional Layer) Lớp tích chập sử dụng các bộ lọc (filters) để trích xuất đặc trưng từ dữ liệu đầu vào. Mỗi bộ lọc là một ma trận nhỏ (ví dụ: 3x3) di chuyển qua dữ liệu đầu vào (như hình ảnh) với một bước trượt (stride) nhất định, thực hiện phép tích chập để tạo ra bản đồ đặc trưng (feature map). Công thức phép tích chập cho một điểm (i, j) trên feature map là:

$$O(i,j) = \sum_{m=0}^{M-1} \sum_{n=0}^{N-1} I(i+m, j+n) \cdot K(m,n)$$

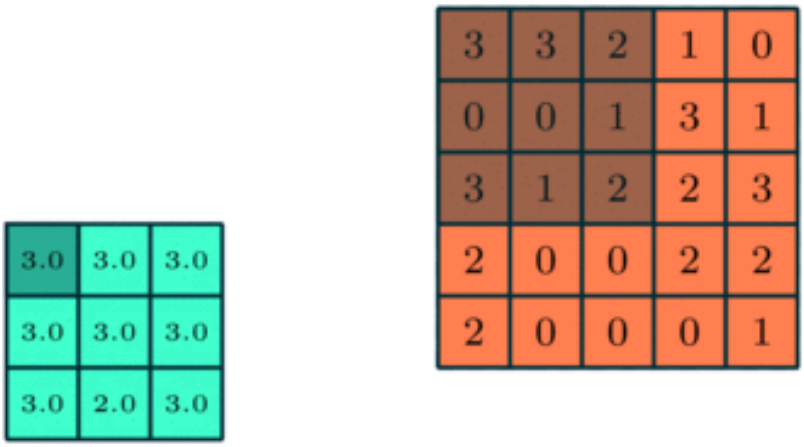
- I: Dữ liệu đầu vào (input), ví dụ: ma trận pixel của hình ảnh.
- K: Ma trận bộ lọc (kernel/filter) kích thước $M \times N$.
- O: Bản đồ đặc trưng đầu ra (feature map).
- (i, j): Vị trí trên feature map.

Lớp gộp (Pooling Layer) Lớp gộp giảm kích thước không gian của feature map, giữ lại thông tin quan trọng và giảm chi phí tính toán. Có hai loại pooling phổ biến:

- **Max Pooling**: Lấy giá trị lớn nhất trong một vùng (ví dụ: 2x2).

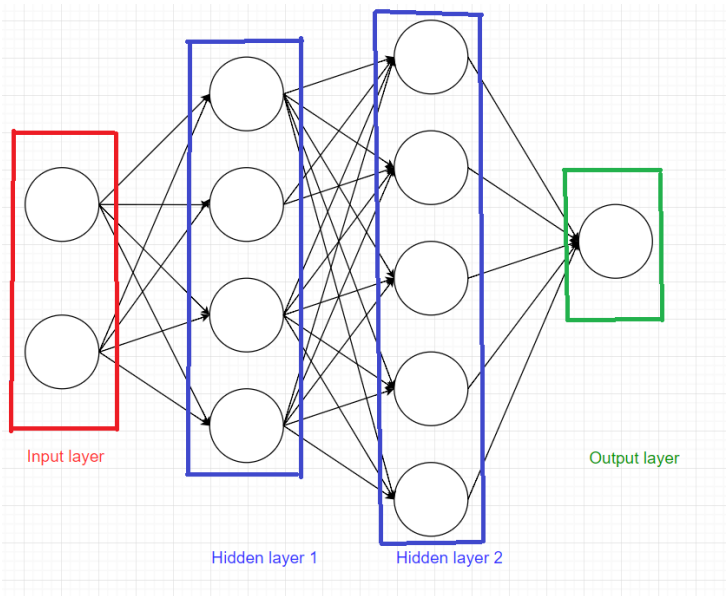
- **Average Pooling**: Lấy giá trị trung bình trong vùng.

Ví dụ, với một feature map 4x4 và Max Pooling 2x2, đầu ra sẽ là một ma trận 2x2 chứa các giá trị lớn nhất từ mỗi vùng 2x2.



Hình 2.3: Ví dụ về pooling [14]

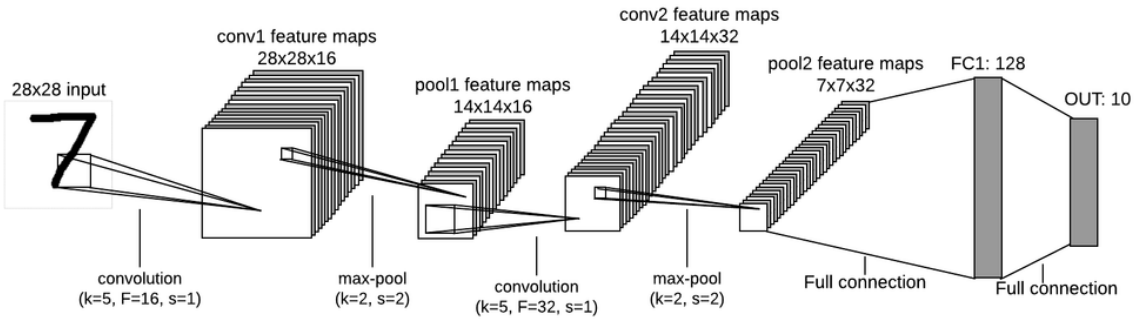
Lớp kết nối đầy đủ (Fully Connected Layer) Sau các lớp tích chập và gộp, dữ liệu được làm phẳng (flatten) thành một vector và đưa vào lớp fully connected. Lớp này kết hợp các đặc trưng để đưa ra dự đoán cuối cùng, thường được sử dụng trong bài toán phân loại. Ví dụ, trong nhận diện hình ảnh, lớp này có thể dự đoán xác suất hình ảnh thuộc các lớp như chó, mèo, hoặc xe hơi.



Hình 2.4: Ví dụ về fully connected [14]

2.2.2 Cách hoạt động của CNN

Dữ liệu đầu vào (như hình ảnh) được đưa qua các lớp tích chập để trích xuất đặc trưng. Mỗi lớp tích chập áp dụng nhiều bộ lọc, tạo ra nhiều feature map. Các lớp gộp giảm kích thước dữ liệu, sau đó dữ liệu được làm phẳng và đưa vào lớp fully connected để phân loại. Quá trình huấn luyện sử dụng lan truyền xuôi (forward propagation) để tính đầu ra và lan truyền ngược (backpropagation) để tối ưu trọng số dựa trên hàm mất mát.



Hình 2.5: Luồng xử lý dữ liệu trong CNN [14]

2.3 Mạng nơ-ron hồi tiếp dài ngắn hạn (LSTM)

Mạng nơ-ron hồi tiếp dài ngắn hạn (Long Short-Term Memory - LSTM) là một loại mạng nơ-ron hồi tiếp (RNN) được thiết kế để xử lý dữ liệu chuỗi (sequential data), như văn bản hoặc chuỗi thời gian, bằng cách khắc phục vấn đề vanishing gradient thường gặp trong RNN truyền thống [6]. LSTM có khả năng học và ghi nhớ các phụ thuộc dài hạn (long-term dependencies) trong chuỗi dữ liệu, rất phù hợp cho các bài toán như xử lý ngôn ngữ tự nhiên (NLP), nhận diện giọng nói, và trả lời câu hỏi trong Visual Question Answering (VQA).

2.3.1 Cơ chế hoạt động của LSTM

LSTM hoạt động dựa trên các cổng (gates) để kiểm soát luồng thông tin qua từng bước thời gian (timestep). Các cổng này bao gồm:

- **Cổng quên (Forget Gate):** Quyết định thông tin nào từ trạng thái ô nhớ C_{t-1} cần loại bỏ. Công thức:
$$f_t = \sigma(W_f \cdot [h_{t-1}, x_t] + b_f)$$
  - f_t: Giá trị cổng quên, trong khoảng $[0, 1]$ (sigmoid).
  - σ: Hàm sigmoid, chuẩn hóa đầu ra.
  - W_f, b_f: Trọng số và bias của cổng quên.
  - h_{t-1}: Đầu ra ẩn thời điểm trước.
  - x_t: Đầu vào hiện tại.
- **Cổng vào (Input Gate):** Quyết định thông tin mới nào được thêm vào trạng thái ô nhớ. Bao gồm:

– Quyết định giá trị cần cập nhật:

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

* i_t: Giá trị cổng vào, trong khoảng $[0, 1]$ (sigmoid).
* W_i, b_i: Trọng số và bias của cổng vào.

– Tạo ứng viên giá trị mới:

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

* $\tilde{C}_t$: Giá trị ứng viên, trong khoảng $[-1, 1]$ (tanh).
* W_C, b_C: Trọng số và bias cho ứng viên.

Trạng thái ô nhớ được cập nhật:

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

– C_t: Trạng thái ô nhớ mới.

– $f_t \cdot C_{t-1}$: Phần thông tin giữ lại.

– $i_t \cdot \tilde{C}_t$: Phần thông tin mới thêm vào.

- **Cổng ra (Output Gate):** Quyết định đầu ra h_t dựa trên trạng thái ô nhớ và đầu vào hiện tại:

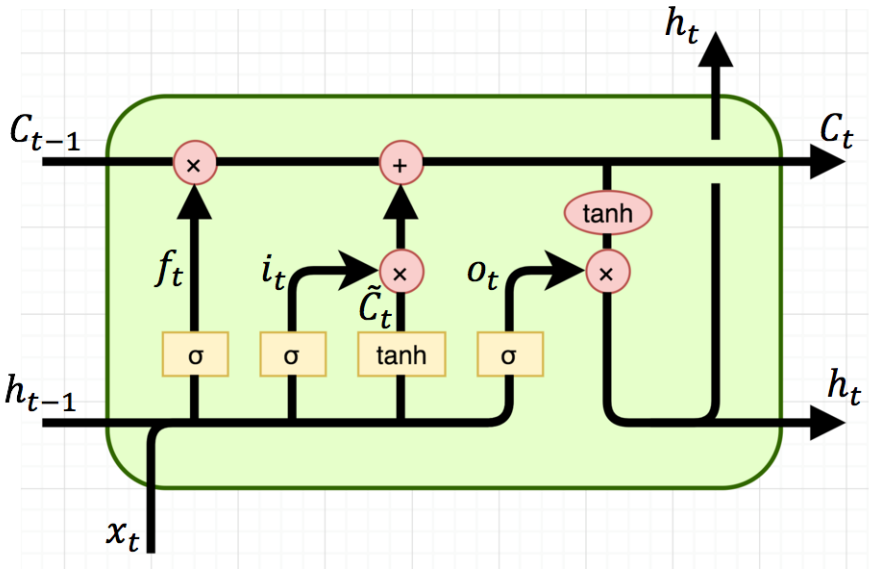
$$o_t = \sigma(W_o \cdot [h_{t-1}, x_t] + b_o)$$
$$h_t = o_t \cdot \tanh(C_t)$$

– o_t: Giá trị cổng ra, trong khoảng $[0, 1]$ (sigmoid).

– W_o, b_o: Trọng số và bias của cổng ra.

– h_t: Đầu ra ẩn hiện tại.

– $\tanh(C_t)$: Trạng thái ô nhớ được chuẩn hóa trong $[-1, 1]$.

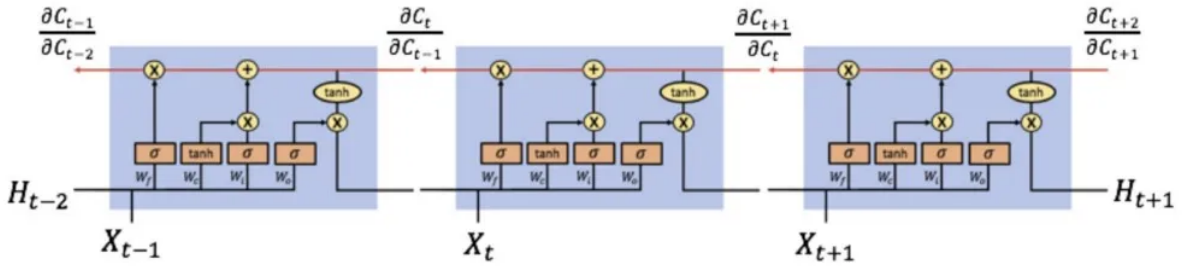


Hình 2.6: Luồng tính toán của các cổng trong LSTM[18]

2.3.2 LSTM khắc phục vấn đề Vanishing Gradient

Vấn đề vanishing gradient trong RNN xảy ra khi gradient trở nên quá nhỏ trong quá trình lan truyền ngược (backpropagation), khiến mạng không thể học hiệu quả các phụ thuộc dài hạn. LSTM giải quyết vấn đề này thông qua:

- **Trạng thái ô nhớ (Cell State):** Cho phép thông tin được truyền qua các bước thời gian mà không bị thay đổi nhiều, trừ khi được điều chỉnh bởi các cổng.
- **Các cổng điều khiển:** Cổng quên và cổng vào giúp mạng quyết định thông tin nào cần giữ lại hoặc bỏ đi, tránh việc gradient bị thu nhỏ quá mức.



Hình 2.7: Minh họa cách LSTM khắc phục vấn đề vanishing gradient thông qua trạng thái ô nhớ[18]

2.4 Tokenization

2.4.1 Các phương pháp Tokenization

- **Word-based Tokenization**: *Word-based tokenization* phân tách văn bản thành các từ riêng lẻ, sử dụng dấu cách hoặc dấu câu làm ranh giới. Ví dụ, câu "Tôi đang học AI" được chia thành ["Tôi", "đang", "học", "AI"]. Phương pháp này đơn giản, giữ được ngữ nghĩa của từ, và phù hợp với các ngôn ngữ như tiếng Anh hoặc tiếng Việt, nơi từ được phân tách rõ ràng . Tuy nhiên, nó gặp khó khăn với từ hiếm, từ mới, hoặc các ngôn ngữ không dùng dấu cách như tiếng Trung, dẫn đến từ điển lớn và khó quản lý.
- **Character-based Tokenization**: *Character-based tokenization* chia văn bản thành các ký tự riêng lẻ, chẳng hạn "Tôi đang học Toán" thành ["T", "ô", "i", " ", "đ", "a", "n", "g", ...]. Phương pháp này tạo ra từ điển nhỏ, xử lý được mọi ngôn ngữ và không gặp vấn đề với từ hiếm. Tuy nhiên, nó làm mất ngữ nghĩa ngữ cảnh của từ và tạo chuỗi token dài, tăng chi phí tính toán, đặc biệt trong các mô hình xử lý chuỗi dài như LSTM hoặc Transformer.
- **Subword-based Tokenization**:*Subword-based tokenization* chia văn bản thành các đơn vị subword, kết hợp ưu điểm của từ và ký tự. Ví dụ, câu Tôi đang học transformer" được chia thành [[CLS]", Tôi", đang", học", trans", ##former", [SEP]"], giúp xử lý từ hiếm hiệu quả với từ điển khoảng 30K token [7]. Phương pháp này hỗ trợ các tác vụ NLP như VQA, nơi văn bản cần được xử lý linh hoạt.

2.4.2 BertTokenizer

BertTokenizer là một công cụ *tokenization* trong thư viện Hugging Face Transformers, được thiết kế cho mô hình BERT, sử dụng phương pháp *Subword-based* [7]. Công cụ này chuyển đổi văn bản thô thành các đơn vị subword, thêm các token đặc biệt như [CLS] và [SEP], sau đó ánh xạ thành token IDs để làm đầu vào cho mô hình. Ví dụ, câu What is a transformer?" được chia thành [[CLS]", What", is", a", trans", ##former", ?", "[SEP]"].

2.5 Bộ tối ưu hóa (Optimizer)

Bộ tối ưu hóa (Optimizer) là thành phần cốt lõi trong quá trình huấn luyện các mô hình học sâu, giúp điều chỉnh trọng số của mạng nơ-ron để giảm thiểu hàm mất mát. Trong bài toán Vision Question Answering (VQA), bộ tối ưu hóa đảm bảo mô hình hội tụ hiệu quả trên tập dữ liệu phức tạp như VQA v2.0. Các bộ tối ưu hóa phổ biến bao gồm Stochastic Gradient Descent (SGD) và Adam, trong đó Adam được sử dụng rộng rãi nhờ khả năng thích nghi với gradient [17].

2.5.1 Gradient Descent và Stochastic Gradient Descent (SGD)

- **Gradient Descent (GD):** Tối ưu hóa hàm mất mát bằng cách cập nhật trọng số dựa trên gradient:
$$w \leftarrow w - \eta \cdot \nabla_w \mathcal{L}$$
  - w: Trọng số của mô hình.
  - η: Tốc độ học (learning rate), điều chỉnh bước cập nhật.
  - $\nabla_w \mathcal{L}$: Gradient của hàm mất mát $\mathcal{L}$ theo w.
- **Stochastic Gradient Descent (SGD):** Dùng mẫu ngẫu nhiên hoặc batch nhỏ để tính gradient, giảm chi phí và tăng tốc hội tụ. Kết hợp momentum để ổn định:
$$v_t = \gamma v_{t-1} + \eta \cdot \nabla_w \mathcal{L}, \quad w \leftarrow w - v_t$$
  - v_t: Vận tốc tại bước t, tích lũy gradient.
  - γ: Hệ số momentum (thường 0.9), kiểm soát ảnh hưởng của vận tốc trước.
  - η: Tốc độ học, tương tự GD.
  - $\nabla_w \mathcal{L}$: Gradient của hàm mất mát $\mathcal{L}$ theo w.
  - w: Trọng số được cập nhật.

2.5.2 Adam Optimizer

Adam (Adaptive Moment Estimation) kết hợp ưu điểm của SGD với momentum và RMSprop, sử dụng trung bình động của gradient và bình phương gradient để điều chỉnh tốc độ học thích nghi [17]. Công thức cập nhật:

$$m_t = \beta_1 m_{t-1} + (1 - \beta_1) \cdot \nabla_w \mathcal{L}, \quad v_t = \beta_2 v_{t-1} + (1 - \beta_2) \cdot (\nabla_w \mathcal{L})^2$$

$$\hat{m}_t = \frac{m_t}{1 - \beta_1^t}, \quad \hat{v}_t = \frac{v_t}{1 - \beta_2^t}, \quad w \leftarrow w - \eta \cdot \frac{\hat{m}_t}{\sqrt{\hat{v}_t} + \epsilon}$$

- m_t: Trung bình động của gradient (first moment).
- v_t: Trung bình động của bình phương gradient (second moment).
- β_1, β_2: Hệ số làm mượt, thường 0.9 và 0.999.
- $\hat{m}_t, \hat{v}_t$: Trung bình động được hiệu chỉnh để giảm bias ban đầu.
- η: Tốc độ học (learning rate).
- $\nabla_w \mathcal{L}$: Gradient của hàm mất mát $\mathcal{L}$ theo trọng số w.
- ϵ: Hằng số nhỏ (thường 10^{-8}) để tránh chia cho 0.
- w: Trọng số được cập nhật.

Trong đề tài, Adam được sử dụng để huấn luyện mô hình VQA, phù hợp với các thành phần như CNN, LSTM, và Knowledge Distillation, nhờ khả năng xử lý gradient phức tạp.

2.6 Lịch trình tốc độ học (Learning Rate Scheduler)

Lịch trình tốc độ học (Learning Rate Scheduler) điều chỉnh tốc độ học η trong quá trình huấn luyện để cải thiện tốc độ hội tụ và hiệu suất mô hình. Trong bài toán VQA, LambdaLR là một phương pháp linh hoạt, cho phép định nghĩa hàm tùy chỉnh để thay đổi tốc độ học theo thời gian hoặc số bước huấn luyện, phù hợp với các mô hình phức tạp như CNN và LSTM [19].

2.6.1 LambdaLR

LambdaLR (Lambda Learning Rate) điều chỉnh tốc độ học bằng cách nhân tốc độ học ban đầu η_0 với một hệ số $\lambda(t)$, được định nghĩa qua một hàm tùy chỉnh phụ thuộc vào bước huấn luyện t hoặc epoch:

$$\eta_t = \eta_0 \cdot \lambda(t)$$

- η_t: Tốc độ học tại bước t.
- η_0: Tốc độ học ban đầu.
- $\lambda(t)$: Hàm hệ số tùy chỉnh, phụ thuộc vào t (bước hoặc epoch).

Hàm $\lambda(t)$ có thể mô phỏng các chiến lược như giảm tuyến tính, giảm hàm mũ, hoặc cosine annealing. Ví dụ, giảm tuyến tính:

$$\lambda(t) = 1 - \frac{t}{T}$$

- t: Bước huấn luyện hiện tại.
- T: Tổng số bước huấn luyện.
- Kết quả: Tốc độ học giảm tuyến tính từ η_0 về 0.

Một dạng khác là cosine decay:

$$\lambda(t) = \frac{1}{2}\left(1 + \cos\left(\frac{t\pi}{T}\right)\right)$$

- Kết quả: Tốc độ học giảm mượt mà, tương tự Cosine Annealing.
- Ứng dụng: Giúp mô hình hội tụ ổn định hơn ở giai đoạn cuối.

2.7 Knowledge Distillation

2.7.1 Giới thiệu và động lực

Knowledge Distillation (KD) là một kỹ thuật nén mô hình được giới thiệu bởi Hinton và cộng sự vào năm 2015 [12]. Kỹ thuật này cho phép một mô hình lớn và mạnh hơn (Teacher Model) truyền tải tri thức ở dạng cô đọng cho một mô hình nhỏ và nhẹ hơn (Student Model), thông qua việc sử dụng đặc trưng ẩn hoặc đầu ra dự đoán [9]. Mục tiêu chính của KD là giảm độ phức tạp tính toán và yêu cầu tài nguyên, đồng thời duy trì hiệu suất cao, rất phù hợp cho các ứng dụng Visual Question Answering (VQA) trên thiết bị có tài nguyên hạn chế như thiết di động. Động lực của KD xuất phát từ nhu cầu triển khai các mô hình học sâu phức tạp trong các môi trường thực tế, nơi mà tài nguyên tính toán và bộ nhớ bị giới hạn [9].

2.7.2 Cơ chế hoạt động của Knowledge Distillation

Quy trình KD bao gồm các thành phần chính như sau:

- **Teacher Model:** Trong đề tài, nhóm nghiên cứu sử dụng ResNet50 (được huấn luyện trước trên ImageNet) để trích xuất đặc trưng ảnh và Bi-LSTM để mã hoá câu hỏi thành đặc trưng ngữ nghĩa, tạo ra các đặc trưng chất lượng cao. Teacher Model thường là một mô hình lớn, đã được huấn luyện tốt, có khả năng cung cấp dự đoán chính xác và đặc trưng phong phú.
- **Student Model:** Student Model là một mô hình nhỏ hơn, trong trường hợp này sử dụng CNN để xử lý hình ảnh và LSTM đơn hướng cho văn bản. Mô hình này được thiết kế để học từ Teacher Model thông qua việc tối ưu hóa một hàm mất mát kết hợp:
  - **Hàm mất mát Cross-Entropy:** Đo lường sự khác biệt giữa dự đoán của Student Model và nhãn thật:

$$\mathcal{L}_{CE} = -\sum_i y_i \log(\hat{y}_i)$$

    * y_i: Nhãn thật (ground truth).
    * $\hat{y}_i$: Xác suất dự đoán của Student Model.
  - **Hàm mất mát Knowledge Distillation (KL Divergence):** Đo lường sự khác biệt giữa xác suất đầu ra của Teacher Model và Student Model, sử dụng tham số nhiệt độ T:

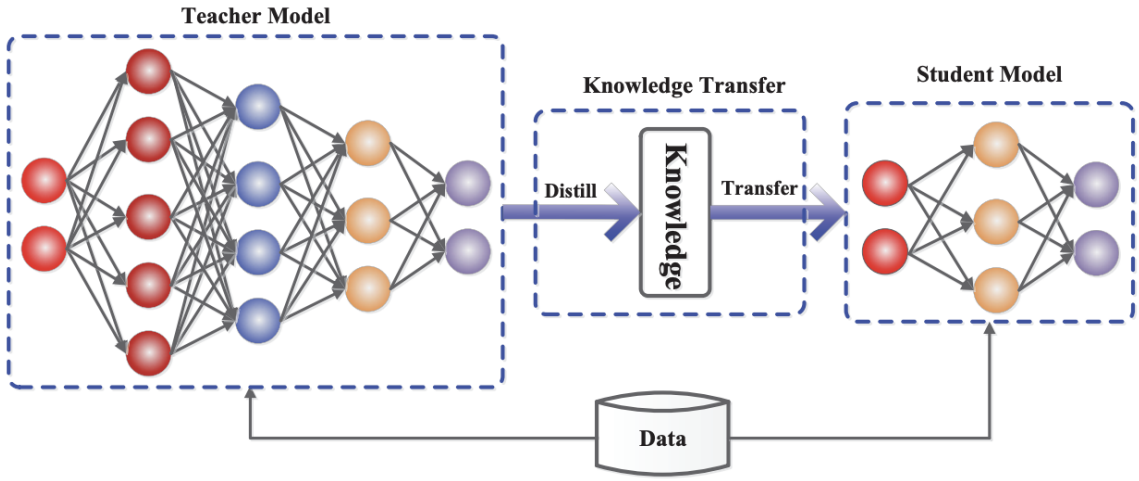
$$\mathcal{L}_{KD} = T^2 \cdot \mathrm{KL}\left(\mathrm{softmax}\left(\frac{\mathrm{logits}_T}{T}\right), \mathrm{softmax}\left(\frac{\mathrm{logits}_S}{T}\right)\right)$$

  - $*$ logits_T: Đầu ra logits của Teacher Model.
  - $*$ logits_S: Đầu ra logits của Student Model.
  - $*$ T: Tham số nhiệt độ ($T > 1$) để làm mềm phân phối xác suất.
  - $*$ KL: Độ đo KL Divergence giữa hai phân phối xác suất.
  - $*$ T^2: Hệ số điều chỉnh độ lớn của mất mát.
- **Hàm mất mát tổng hợp:** Kết hợp hai thành phần trên:

$$\mathcal{L} = \alpha \cdot \mathcal{L}_{CE} + (1 - \alpha) \cdot \mathcal{L}_{KD}$$

  - $*$ α: Tham số cân bằng giữa $\mathcal{L}_{CE}$ và $\mathcal{L}_{KD}$.
  - $*$ $\mathcal{L}_{CE}$: Mất mát Cross-Entropy.
  - $*$ $\mathcal{L}_{KD}$: Mất mát Knowledge Distillation.

- **Huấn luyện:** Student Model được huấn luyện để mô phỏng đầu ra của Teacher Model (soft targets) và đồng thời học từ dữ liệu thực tế (hard targets). Quá trình này cho phép Student Model không chỉ học từ nhãn mà còn từ phân phối xác suất phong phú của Teacher Model, giúp cải thiện khả năng tổng quát hóa [12].



Hình 2.8: Quy trình Knowledge Distillation: Teacher Model truyền kiến thức cho Student Model thông qua hàm mất mát kết hợp[9]

2.7.3 Các phương pháp Knowledge Distillation

Theo [9], tri thức trong KD được truyền tải thông qua ba dạng chính, mỗi dạng lại cung cấp một cách độc đáo để khai thác và truyền tải thông tin:

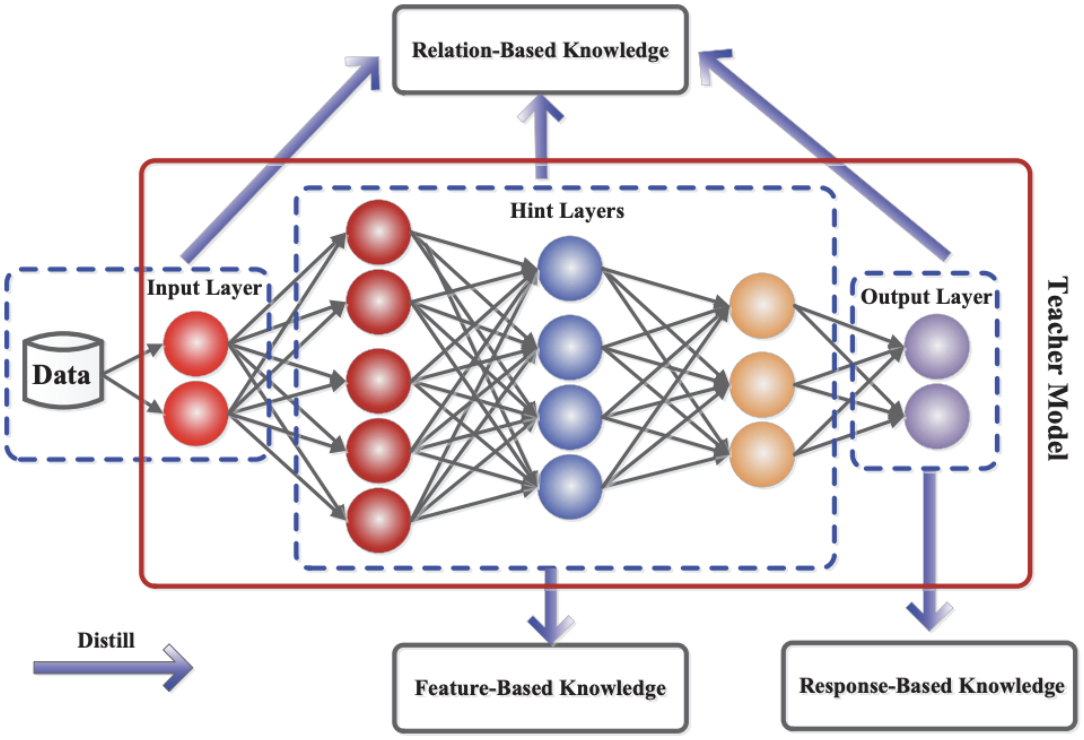
- **Response-based Knowledge:** Mô phỏng đầu ra cuối cùng (logits hoặc xác suất) của Teacher Model, theo phương pháp của Hinton et al. Sử dụng hàm mất mát KL Divergence.
  - Mục tiêu: Student Model học phân phối xác suất đầu ra của Teacher.

- **Feature-based Knowledge:** Student Model học từ đặc trưng trung gian (bản đồ đặc trưng) của Teacher Model. Sử dụng Mean Squared Error (MSE):

$$\mathcal{L}_{MSE} = \frac{1}{N}\sum_{i=1}^{N}(f_T^i - f_S^i)^2$$

  - f_T^i: Đặc trưng ẩn của Teacher Model tại tầng i.
  - f_S^i: Đặc trưng ẩn của Student Model tại tầng i.
  - N: Số lượng đặc trưng.
  - $\mathcal{L}_{MSE}$: Đo sự khác biệt giữa đặc trưng của Teacher và Student.

- **Relation-based Knowledge:** Học các mối quan hệ giữa đặc trưng, như tương quan giữa cặp đặc trưng hoặc giữa các tầng.
  - Mục tiêu: Bắt chước cấu trúc quan hệ trong Teacher Model.



Hình 2.9: Minh họa các phương pháp Knowledge Distillation: Response-based, Feature-based, và Relation-based[9]

2.7.4 Các biến thể của Knowledge Distillation

KD đã được phát triển thành nhiều biến thể để cải thiện hiệu quả:

- **Online KD:** Teacher và Student được huấn luyện đồng thời, thay vì Teacher được huấn luyện trước. Điều này hữu ích khi không có sẵn Teacher Model đã huấn luyện tốt.

- **Multi-Teacher KD:** Sử dụng nhiều Teacher Model để hướng dẫn một Student Model, tận dụng kiến thức đa dạng từ các nguồn khác nhau.
- **Adversarial KD:** Kết hợp KD với các kỹ thuật đối kháng (adversarial learning) để cải thiện khả năng học của Student Model.

2.7.5 Ưu điểm, nhược điểm và thách thức

Ưu điểm:

- KD cho phép xây dựng các mô hình nhẹ hơn, phù hợp với các ứng dụng thực tế trên thiết bị hạn chế tài nguyên, như trong VQA.
- Tận dụng được kiến thức từ các mô hình lớn, giúp Student Model đạt hiệu suất cao hơn so với huấn luyện độc lập.
- Linh hoạt, có thể áp dụng trong nhiều lĩnh vực như xử lý hình ảnh, NLP, và nhận diện giọng nói [9].

Nhược điểm:

- Hiệu suất của Student Model phụ thuộc lớn vào chất lượng của Teacher Model và cách thiết kế hàm mất mát.
- Sự khác biệt về kiến trúc giữa Teacher và Student (ví dụ: ResNet50 so với CNN tùy chỉnh) có thể gây khó khăn trong việc truyền tải kiến thức hiệu quả.
- Quá trình huấn luyện có thể phức tạp, đặc biệt khi sử dụng các biến thể như Multi-Teacher KD hoặc Adversarial KD.

Thách thức:

- **Khoảng cách kiến trúc:** Sự khác biệt giữa Teacher và Student có thể dẫn đến mất mát thông tin trong quá trình truyền tải kiến thức.
- **Lựa chọn tham số:** Việc chọn tham số nhiệt độ T và hệ số α trong hàm mất mát đòi hỏi thử nghiệm cẩn thận.
- **Tổng quát hóa:** Student Model có thể gặp khó khăn trong việc tổng quát hóa nếu Teacher Model bị overfitting hoặc không phù hợp với dữ liệu mục tiêu.

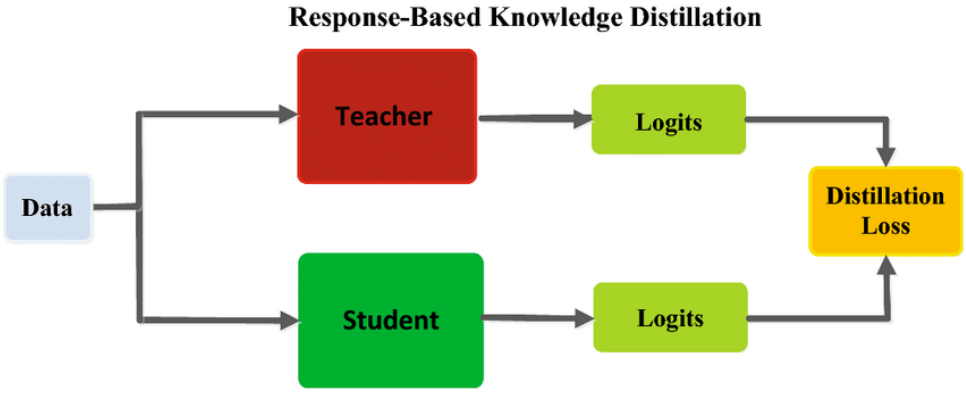
2.8 Ứng dụng các phương pháp trong bài toán VQA

Các phương pháp trên được tích hợp để giải quyết bài toán VQA như sau:

- **CNN và Pre-trained ResNet50:** Trích xuất đặc trưng hình ảnh từ tập dữ liệu VQA v2.0, cung cấp thông tin không gian cho mô hình.
- **LSTM và Bi-LSTM:** Xử lý câu hỏi đã được mã hóa bởi BERT Tokenizer, tạo ra đặc trưng ngữ nghĩa.
- **BERT Tokenizer:** Mã hóa câu hỏi thành các token subword, đảm bảo đầu vào đồng nhất cho LSTM.

- **KD:** Tối ưu hóa mô hình Student (CNN + LSTM) bằng cách học từ mô hình Teacher (Pre-trained ResNet50 + Bi-LSTM), giảm chi phí tính toán và duy trì hiệu suất.

Sự kết hợp này đảm bảo hệ thống VQA có khả năng trả lời câu hỏi Yes/No với độ chính xác cao, đồng thời khả thi về mặt triển khai thực tế [8].



Hình 2.10: Pipeline mô phỏng hướng tiếp cận theo Response-based Knowledge trong KD[9]

2.9 Tổng kết chương

Kết chương này đã trình bày một cách chi tiết các cơ sở lý thuyết cần thiết cho bài toán VQA, bao gồm khái niệm, cơ chế hoạt động, và các ưu nhược điểm của các công nghệ như CNN, LSTM, BERT Tokenizer, và Knowledge Distillation (KD). Những vấn đề đã đặt ra trong phần Tổng quan, từ việc tích hợp xử lý hình ảnh và văn bản đến tối ưu hóa mô hình, đã được giải quyết thông qua việc phân tích các kỹ thuật học sâu và phương pháp KD, cung cấp nền tảng vững chắc để triển khai hệ thống VQA trong thực tế. Các công nghệ này không chỉ hỗ trợ việc trích xuất đặc trưng mà còn giúp giảm độ phức tạp tính toán, tạo tiền đề cho việc xây dựng và đánh giá mô hình VQA trong chương tiếp theo. Chương 3 sẽ trình bày chi tiết quá trình thực nghiệm và đánh giá, từ thiết lập dữ liệu, xây dựng mô hình, đến triển khai ứng dụng thực tiễn.

3 Thực nghiệm và đánh giá

Phần này trình bày thiết lập thực nghiệm và kết quả đánh giá mô hình Vision Question Answering (VQA) dạng Yes/No, sử dụng tập dữ liệu VQA v2.0. Các thí nghiệm được thiết kế để đánh giá hiệu quả của mô hình học sinh được huấn luyện thông qua Knowledge Distillation (KD) từ mô hình giáo viên, cùng với so sánh các phương pháp khác nhau. nhóm nghiên cứu tập trung vào phương pháp xây dựng mô hình, tham số, chiến lược tối ưu hóa, và phân tích hiệu suất trên các tập huấn luyện, xác thực, và kiểm tra. Phần cuối trình bày triển khai ứng dụng thực tế bằng Streamlit.

3.1 Thiết lập thực nghiệm

3.1.1 Tập dữ liệu

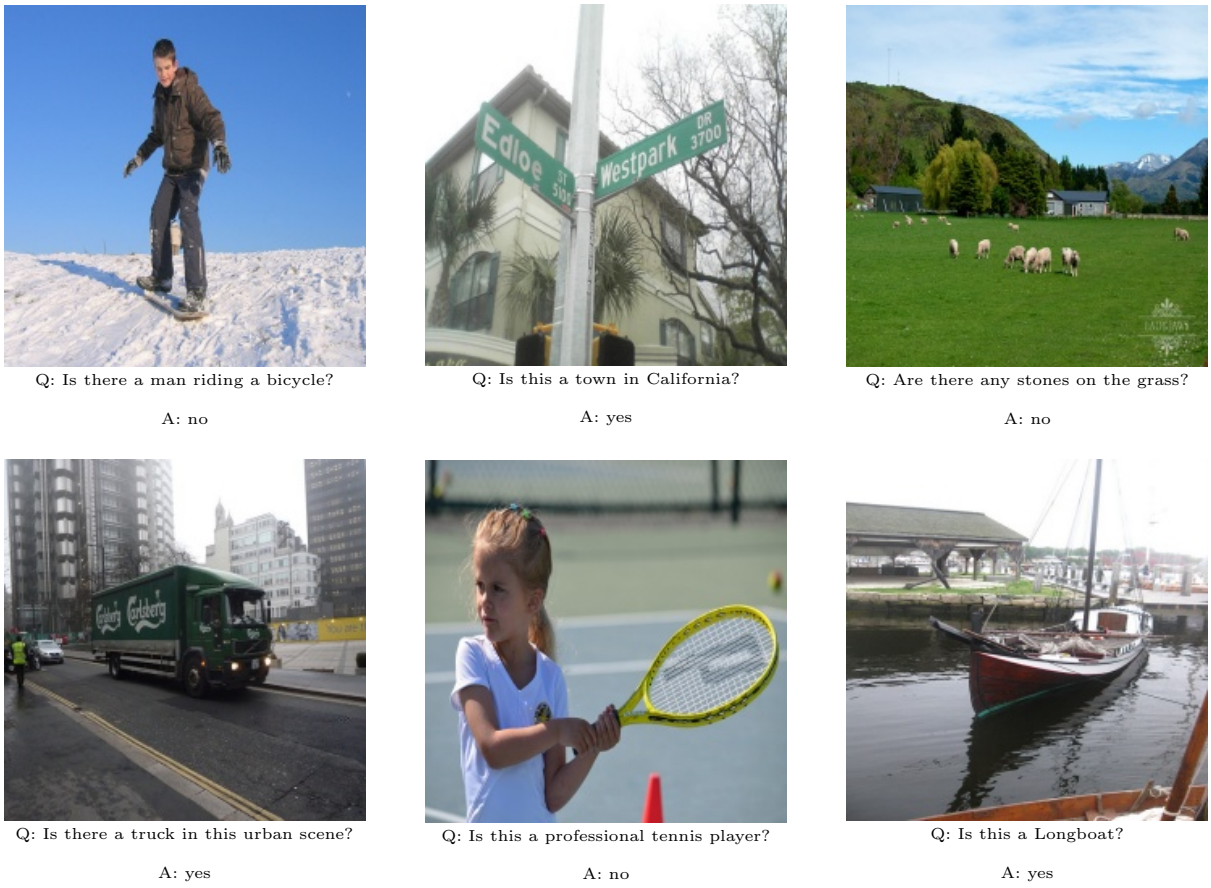
Tập dữ liệu VQA v2.0 [1], được tải từ Kaggle, bao gồm các hình ảnh từ tập COCO và các cặp câu hỏi-đáp dạng Yes/No. Dữ liệu được chia thành ba tập: huấn luyện (44,337 mẫu), xác thực (21,465 mẫu), và kiểm tra (12,576 mẫu), với cấu trúc:

- **Hình ảnh**: Lưu trữ trong thư mục `val2014-resized`, kích thước 224x224 pixel.
- **Câu hỏi và câu trả lời**: Lưu trong các tệp `.txt` (ví dụ: `vaq2.0.TrainImages.txt`), định dạng gồm tên hình ảnh, câu hỏi, và nhãn (`yes` hoặc `no`).

Tập dữ liệu	Số mẫu	Tỷ lệ Yes (%)	Tỷ lệ No (%)
Huấn luyện	44,337	52.1	47.9
Xác thực	21,465	51.8	48.2
Kiểm tra	12,576	50.9	49.1

Bảng 3.1: Thống kê tập dữ liệu VQA v2.0[1]

Phân bố nhãn tương đối cân bằng (Bảng 3.1) giảm nguy cơ thiên lệch, nhưng sự khác biệt về ngữ cảnh hình ảnh và câu hỏi giữa các tập có thể ảnh hưởng đến khả năng tổng quát hóa [8].



Hình 3.1: Một số hình ảnh từ folder val2014-resised[1]

3.1.2 Môi trường tính toán

Thí nghiệm được thực hiện trên Google Colab (NVIDIA Tesla T4 16GB). Môi trường sử dụng PyTorch 1.9.0, CUDA 11.1, cùng các thư viện: `timm` 0.4.12, `transformers` 4.10.0, `torchvision` 0.10.0. Mã nguồn được viết bằng Python 3.8, với seed ngẫu nhiên (`torch.manual_seed(42)`) để đảm bảo tái hiện. Dữ liệu được lưu trữ trên Google Drive, tối ưu hóa truy cập và chia sẻ.

3.2 Tiền xử lý dữ liệu

Dữ liệu được tiền xử lý để đảm bảo tính đồng nhất và phù hợp với các mô hình học sâu:

- **Hình ảnh**: Tải từ thư mục chứa hình ảnh COCO, áp dụng biến đổi:
  - **Tập huấn luyện**: Thay đổi kích thước ảnh về 224x224, gia tăng độ đa dạng bằng cách cắt trung tâm (180x180), lật ngang ngẫu nhiên (xác suất 0.5), làm mờ Gaussian (kernel 3x3), điều chỉnh màu sắc (brightness, contrast, saturation với hệ số 0.1), chuẩn hóa với trung bình (0.485, 0.456, 0.406) và độ lệch chuẩn (0.229, 0.224, 0.225).
  - **Tập xác thực/kiểm tra**: Chỉ thay đổi kích thước và chuẩn hóa.
- **Câu hỏi**: Mã hóa bằng BERT Tokenizer (`bert-base-uncased`) với độ dài tối đa

20 token, thêm token đặc biệt (`[CLS]`, `[SEP]`, `[PAD]`):

$$\text{Encoding}(q) = \texttt{BERTTokenizer(q, max\_length=20, padding='max\_length', truncation=True)}$$

- **Nhãn**: Ánh xạ: yes → 0, no → 1, tạo hai lớp {0: 'yes', 1: 'no'}.

Phân tích: Tăng cường dữ liệu giúp mô hình học đặc trưng mạnh mẽ, nhưng làm mờ Gaussian cần được điều chỉnh để tránh che khuất chi tiết hình ảnh. BERT Tokenizer giữ ngữ nghĩa câu hỏi, nhưng độ dài tối đa 20 token có thể cắt bớt câu hỏi dài, dù phù hợp với câu hỏi Yes/No ngắn. Ánh xạ nhãn đơn giản hóa tính toán mất mát, hỗ trợ phân loại nhị phân hiệu quả [8].

3.3 Chuẩn bị dữ liệu PyTorch

Dữ liệu được tổ chức thành lớp `VQADataset`, kế thừa từ `torch.utils.data.Dataset`, để tải hình ảnh, câu hỏi, và nhãn theo batch. Phương pháp bao gồm:

- **Biến đổi dữ liệu**:
  - **Tập huấn luyện**: Áp dụng chuỗi biến đổi gồm thay đổi kích thước về 224x224, cắt trung tâm (180x180), lật ngang ngẫu nhiên (xác suất 0.5), điều chỉnh màu sắc (brightness, contrast, saturation với hệ số 0.1), làm mờ Gaussian (kernel 3x3, sigma mặc định 0.1–2.0), chuẩn hóa với trung bình (0.485, 0.456, 0.406) và độ lệch chuẩn (0.229, 0.224, 0.225).
  - **Tập xác thực/kiểm tra**: Chỉ thay đổi kích thước và chuẩn hóa để bảo toàn đặc trưng gốc.
  - **Trực quan hóa**: Sử dụng biến đổi tối thiểu (thay đổi kích thước về 224x224, chuyển thành tensor), không chuẩn hóa để giữ màu sắc gốc cho mục đích hiển thị.
- **Cấu hình tải dữ liệu**: Các `DataLoader` được thiết lập với:
  - **Tập huấn luyện**: Batch size 256, shuffle.
  - **Tập xác thực/kiểm tra**: Batch size 64, không shuffle.
  - **Trực quan hóa**: Batch size 64, không shuffle, dùng cho kiểm tra trực quan.
- **Tham số chính**:
  - Độ dài chuỗi tối đa (`max_seq_len`): 20 token.
  - Thư mục hình ảnh (`img_dir`): `val2014-resized`.

Thành phần	Tham số	Giá trị
VQADataset	Độ dài chuỗi tối đa (`max_seq_len`)	20
	Thư mục hình ảnh (`img_dir`)	`val2014-resized`
DataLoader	Batch size (huấn luyện)	256
	Batch size (xác thực/kiểm tra)	64
	Shuffle (huấn luyện)	True
	Shuffle (xác thực/kiểm tra)	False

Bảng 3.2: Tham số của lớp VQADataset và DataLoader [2]

3.4 Xây dựng mô hình

3.4.1 Mô hình giáo viên

Mô hình giáo viên tích hợp bộ mã hóa hình ảnh và văn bản để trích xuất đặc trưng chất lượng cao, đóng vai trò hướng dẫn trong Knowledge Distillation (KD). Phương pháp bao gồm:

- **Bộ mã hóa hình ảnh**: ResNet50 pre-trained trên ImageNet [4], với tầng fully connected cuối được điều chỉnh để xuất đặc trưng kích thước 256. Trọng số pre-trained cung cấp kiến thức ban đầu mạnh mẽ về các mẫu không gian (đối tượng, màu sắc, vị trí).
- **Bộ mã hóa văn bản**: Tầng nhúng chuyển token từ BERT Tokenizer (`vocab_size` = 30,522) thành vector kích thước 128, sử dụng khởi tạo trọng số mặc định (He Initialization). Bi-LSTM hai chiều với 2 tầng, kích thước ẩn 256, dropout 0.15, xuất đặc trưng kích thước 512 (do `bidirect=True`), lấy đầu ra token cuối để đại diện ngữ nghĩa câu hỏi [6].
- **MLP**: Kết hợp đặc trưng hình ảnh (256) và văn bản (512) thành vector 768, giảm chiều qua ba tầng tuyến tính:

$$\text{MLP} : 768 \to 512 \to 128 \to 2, \quad \text{ReLU}, \text{Dropout}(0.15)$$

  Đặc trưng trung gian từ tầng thứ hai (kích thước 128) được sử dụng cho KD, cân bằng giữa thông tin và độ phức tạp.

Tham số chính được liệt kê trong Bảng 3.3.

Tham số	Giá trị
Kích thước ẩn (`hidden_size`)	256
Kích thước nhúng (`embedding_dim`)	128
Kích thước từ điển (`vocab_size`)	30,522
Số tầng LSTM (`n_layers`)	2
Tỷ lệ dropout (`drop_p`)	0.15
Kích thước tầng trung gian (`proj_dim`)	128
LSTM hai chiều (`bidirect`)	True
Số lớp (`n_classes`)	2
Mô hình hình ảnh (`img_model_name`)	ResNet50
Pre-trained	True (ImageNet)

Bảng 3.3: Tham số của mô hình giáo viên[2]

Phân tích:

- **Mục đích**: Tạo mô hình mạnh mẽ để trích xuất đặc trưng chất lượng cao từ hình ảnh và văn bản, cung cấp kiến thức hướng dẫn cho mô hình học sinh trong KD, đảm bảo hiệu suất cao với chi phí tính toán thấp hơn.

- **Ý nghĩa**: ResNet50 pre-trained khai thác kiến thức từ ImageNet, giảm thời gian huấn luyện và cải thiện trích xuất đặc trưng hình ảnh. Bi-LSTM hai chiều với `hidden_size=256` và `n_layers=2` tối ưu hóa hiểu ngữ cảnh câu hỏi, vượt trội so với LSTM đơn hướng trong mô hình học sinh. Dropout 0.15 giảm overfitting, nhưng vẫn cần tinh chỉnh trên dữ liệu VQA để tránh phụ thuộc quá mức vào đặc trưng ImageNet [8]. MLP tích hợp hiệu quả hai luồng dữ liệu, với tầng trung gian (`proj_dim=128`) được chọn cho KD để cân bằng thông tin và độ phức tạp, hỗ trợ mô hình học sinh học các biểu diễn cấp cao. Độ phức tạp của ResNet50 và Bi-LSTM đòi hỏi tối ưu hóa trên Colab, như mixed precision training hoặc gradient clipping, để ổn định huấn luyện trên GPU T4/K80. Khởi tạo trọng số mặc định (Xavier) đảm bảo ổn định, nhưng có thể cải thiện bằng khởi tạo He cho ReLU. Thay thế Bi-LSTM bằng BERT hoặc tích hợp ViT có thể tăng hiệu suất, đặc biệt trong kịch bản few-shot.

3.4.2 Mô hình học sinh

Mô hình học sinh được thiết kế để giảm độ phức tạp tính toán so với mô hình giáo viên, tận dụng Knowledge Distillation (KD) để học các đặc trưng chất lượng cao. Phương pháp bao gồm:

- **Bộ mã hóa hình ảnh**: CNN tùy chỉnh với một tầng tích chập ban đầu (kernel 7x7, stride 2, padding 3, kênh đầu ra 64) kết hợp chuẩn hóa batch và ReLU, tiếp theo là max pooling (kernel 3x3, stride 2, padding 1). Bốn tầng tích chập chính (`layer1`–`layer4`), mỗi tầng gồm hai khối tích chập (kernel 3x3, stride 1 hoặc 2, padding 1), chuẩn hóa batch, ReLU, và kết nối tắt (residual connections). Mỗi khối sử dụng hai tầng tích chập với số kênh lần lượt là 64, 128, 256, 512. Đầu ra được xử lý qua adaptive average pooling (kích thước 1x1) và tầng tuyến tính để tạo đặc trưng kích thước 64. Khởi tạo trọng số sử dụng phương pháp He mặc định cho các tầng tích chập và tuyến tính.

- **Bộ mã hóa văn bản**: Tầng nhúng chuyển token từ BERT Tokenizer (`vocab_size` = 30,522) thành vector kích thước 32, sử dụng khởi tạo He. LSTM đơn hướng với 2 tầng, kích thước ẩn 64, dropout 0.0, xuất đặc trưng kích thước 64, lấy đầu ra token cuối để đại diện ngữ nghĩa câu hỏi.
- **MLP**: Kết hợp đặc trưng hình ảnh (64) và văn bản (64) thành vector 128, giảm chiều qua hai tầng tuyến tính:

$$\text{MLP} : 128 \to 32 \to 2, \quad \text{ReLU}$$

  Tầng tuyến tính sử dụng khởi tạo He và không áp dụng dropout.
- **Regressor**: Tầng tuyến tính $128 \to 128$ mô phỏng đặc trưng trung gian của mô hình giáo viên (kích thước 128) trong KD, sử dụng khởi tạo He.

Tham số chính được liệt kê trong Bảng 3.4.

Tham số	**Giá trị**
Kích thước ẩn (`hidden_size`)	64
Kích thước nhúng (`embedding_dim`)	32
Kích thước từ điển (`vocab_size`)	30,522
Số tầng LSTM (`n_layers`)	2
Tỷ lệ dropout (`drop_p`)	0.0
Kích thước tầng trung gian (`proj_dim`)	32
LSTM hai chiều (`bidirect`)	False
Số lớp (`n_classes`)	2
Kênh đầu vào CNN (`in_channels`)	3
Kênh đầu ra CNN (`out_channels`)	64, 128, 256, 512
Số khối mỗi tầng (`blocks`)	2
Kích thước kernel tích chập	3x3, 7x7 (ban đầu)
Kích thước kernel max pooling	3x3
Stride max pooling	2
Padding max pooling	1
Kích thước đầu ra pooling (`avgpool`)	1x1
Khởi tạo trọng số	He (Conv2d, Linear), Uniform (Embedding)

Bảng 3.4: Tham số của mô hình học sinh[2]

Hàm mất mát KD:

$$\mathcal{L} = \alpha \cdot \mathcal{L}_{\text{CE}}(\hat{y}_S, y) + (1 - \alpha) \cdot \mathcal{L}_{\text{MSE}}(f_S, f_T), \quad \alpha = 0.75 \tag{1}$$

Giải thích các tham số:

- $\mathcal{L}$: Hàm mất mát tổng hợp, kết hợp hai thành phần để tối ưu hóa mô hình học sinh, giúp học từ nhãn thực tế và kiến thức của mô hình giáo viên.
- α: Hệ số cân bằng, ở đây $\alpha = 0.75$, điều chỉnh mức độ ảnh hưởng của $\mathcal{L}_{\text{CE}}$ (75%) và $\mathcal{L}_{\text{MSE}}$ (25%).

- $\mathcal{L}_{\text{CE}}(\hat{y}_S, y)$: Hàm mất mát Cross-Entropy, đo sự khác biệt giữa dự đoán $\hat{y}_S$ (xác suất từ mô hình học sinh) và nhãn thực tế y (0: Yes, 1: No).
- $\mathcal{L}_{\text{MSE}}(f_S, f_T)$: Hàm mất mát Mean Squared Error, đo sự khác biệt giữa đặc trưng trung gian f_S (mô hình học sinh) và f_T (mô hình giáo viên), cả hai có kích thước 128.
- $1-\alpha$: Hệ số cho $\mathcal{L}_{\text{MSE}}$, ở đây $1-\alpha = 0.25$, xác định mức độ đóng góp của việc học từ đặc trưng trung gian.

3.5 Huấn luyện

Mô hình học sinh được huấn luyện trong 50 epoch với:

- **Optimizer**: Adam [17], $\eta = 0.001$, $\beta_1 = 0.9$, $\beta_2 = 0.999$, $\epsilon = 10^{-8}$.
- **Scheduler**: LambdaLR [19]:

$$\lambda(t) = \begin{cases} 0.1 + 0.9 \cdot \frac{t}{5}, & t < 5 \\ \frac{50-t}{45}, & t \geq 5 \end{cases}$$

- **Loss weights**: $\alpha = 0.75$ ($\mathcal{L}_{\text{CE}}$), $1-\alpha = 0.25$ ($\mathcal{L}_{\text{MSE}}$).

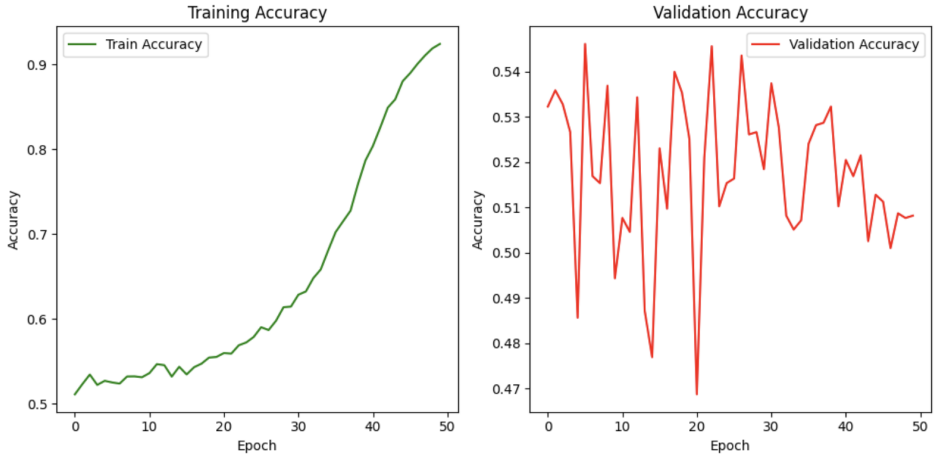
Giải thích nguồn gốc các con số:

- **Optimizer Adam:** Các tham số $\eta = 0.001$, $\beta_1 = 0.9$, $\beta_2 = 0.999$, $\epsilon = 10^{-8}$ được chọn dựa trên gợi ý từ bài báo gốc của Adam [17], trong đó đây là các giá trị mặc định được khuyến nghị để đảm bảo hội tụ ổn định và hiệu quả trên các bài toán học sâu.
- **Scheduler LambdaLR:** Công thức $\lambda(t)$ được thiết kế dựa trên chiến lược điều chỉnh tốc độ học trong [19], với các hằng số 0.1, 0.9, 5, 50, và 45 được nhóm lựa chọn để tăng tốc độ học tuyến tính trong 5 epoch đầu (từ 0.1 lên 1) và giảm dần trong 45 epoch còn lại, phù hợp với tổng số 50 epoch huấn luyện.
- **Loss weights:** Giá trị $\alpha = 0.75$ và $1-\alpha = 0.25$ được nhóm thử nghiệm và chọn dựa trên hiệu suất ban đầu trên tập dữ liệu VQA v2.0, nhằm cân bằng giữa học từ nhãn thực tế ($\mathcal{L}_{\text{CE}}$) và học từ mô hình giáo viên ($\mathcal{L}_{\text{MSE}}$), như được đề xuất trong các nghiên cứu về Knowledge Distillation [12].

Thời gian huấn luyện: **1 giờ 36 phút 55 giây** trên GPU Colab. Kết quả được ghi nhận trong Bảng 3.5.

Epoch	Train Accuracy	Validation Accuracy
1	0.5110	0.5323
2	0.5229	0.5359
3	0.5343	0.5328
10	0.5311	0.4944
20	0.5551	0.5251
30	0.6145	0.5184
40	0.7869	0.5102
50	0.9244	0.5082

Bảng 3.5: Kết quả huấn luyện qua các epoch tiêu biểu[2]



Hình 3.2: Độ chính xác huấn luyện và xác thực qua 50 epoch[2]

Phân tích: Độ chính xác huấn luyện tăng từ 0.5110 lên 0.9244, chứng minh hiệu quả KD. Độ chính xác xác thực dao động, đạt đỉnh 0.5359 (epoch 2) nhưng giảm xuống 0.5082 (epoch 50), do không đồng nhất dữ liệu, thiếu regularization, và α chưa tối ưu [8]. Trong 3 epoch đầu, độ chính xác huấn luyện tăng trung bình 0.0117 mỗi epoch, xác thực chỉ tăng 0.0002, cho thấy khó khăn trong tổng quát hóa sớm.

3.6 Đánh giá

Mô hình được đánh giá bằng độ chính xác phân loại:

$$\text{Accuracy} = \frac{\sum_{i=1}^{N} \mathbb{I}(\hat{y}_i = y_i)}{N} \tag{2}$$

Giải thích các tham số:

- Accuracy: Độ chính xác phân loại, biểu thị tỷ lệ các dự đoán đúng trên tổng số mẫu.
- $\sum_{i=1}^{N} \mathbb{I}(\hat{y}_i = y_i)$: Tổng số dự đoán đúng, trong đó $\mathbb{I}(\hat{y}_i = y_i)$ là hàm chỉ báo, bằng 1 nếu dự đoán $\hat{y}_i$ khớp với nhãn thực tế y_i, và 0 nếu không khớp.

- $\hat{y}_i$: Dự đoán của mô hình cho mẫu thứ i, ở đây là nhãn Yes/No (0 hoặc 1).
- y_i: Nhãn thực tế của mẫu thứ i, cũng là nhãn Yes/No (0 hoặc 1).
- N: Tổng số mẫu được đánh giá (ví dụ: 21,465 mẫu cho tập xác thực, 12,576 mẫu cho tập kiểm tra).

Kết quả của phương pháp gốc:

- **Validation accuracy**: 0.5082
- **Test accuracy**: 0.5479

Nhận xét về kết quả: Kết quả cho thấy những dấu hiệu tích cực ban đầu, với độ chính xác trên tập kiểm tra đạt 54.79%, cao hơn một chút so với tập xác thực (50.82%) và vượt mức ngẫu nhiên (50%) trong bài toán phân loại nhị phân Yes/No. Đây là một bước tiến khả quan, cho thấy tiềm năng của mô hình trong việc áp dụng thực tế, mặc dù hiện tượng overfitting cho thấy vẫn còn không gian để cải thiện khả năng tổng quát hóa.

Giải thích nguyên nhân:

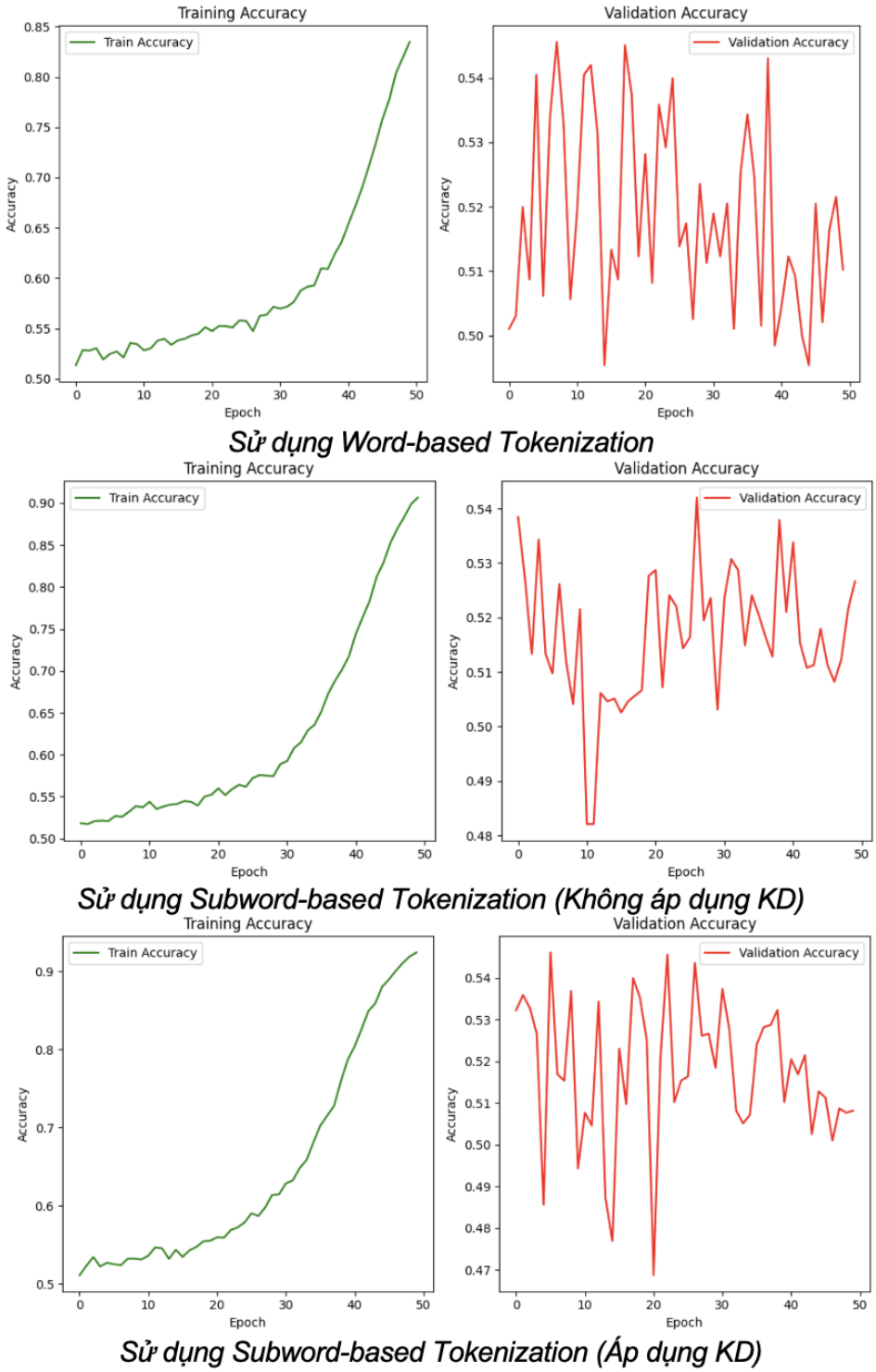
- **Tiềm năng của KD:** Việc áp dụng Knowledge Distillation (KD) đã giúp độ chính xác huấn luyện tăng mạnh lên 92.44% (Bảng 3.5), chứng minh hiệu quả của phương pháp trong việc học từ mô hình giáo viên, tạo nền tảng tốt cho các cải tiến tiếp theo.
- **Độ chính xác kiểm tra cao hơn xác thực:** Độ chính xác kiểm tra (54.79%) cao hơn xác thực (50.82%) cho thấy mô hình có khả năng thích nghi tốt hơn trên dữ liệu mới, một tín hiệu tích cực cho việc triển khai thực tế trên Streamlit.
- **Overfitting như một cơ hội học hỏi:** Độ chính xác huấn luyện cao (92.44%) nhưng xác thực/kiểm tra thấp hơn (50.82% và 54.79%) cho thấy mô hình bị overfitting, mở ra cơ hội áp dụng các kỹ thuật regularization để cải thiện khả năng tổng quát hóa [8].
- **Cơ hội cải thiện từ dữ liệu:** Sự khác biệt về ngữ cảnh giữa các tập dữ liệu (huấn luyện, xác thực, kiểm tra) là một cơ hội để tinh chỉnh mô hình, giúp tăng khả năng tổng quát hóa trong tương lai.

3.6.1 So sánh các phương pháp

Để đánh giá hiệu quả của phương pháp sử dụng Subword-based Tokenization, nhóm nghiên cứu tiến hành so sánh với hai phương pháp khác là Word-based Tokenization [2] và Subword-based Tokenization (Không có KD) [2] , nhằm cải thiện khả năng tổng quát hóa của mô hình. Kết quả so sánh được thể hiện trong bảng và được trực quan hóa thông qua hình sau.

Phương pháp	Train Accuracy	Validation Accuracy
Word-based Tokenization	0.8348	0.5102
Subword-based Tokenization (Không dùng KD)	0.9064	0.5266
Subword-based Tokenization (Áp dụng KD)	0.9244	0.5082

Bảng 3.6: So sánh độ chính xác của các phương pháp[2]



Hình 3.3: Biểu đồ so sánh độ chính xác huấn luyện và xác thực của các phương pháp[2]

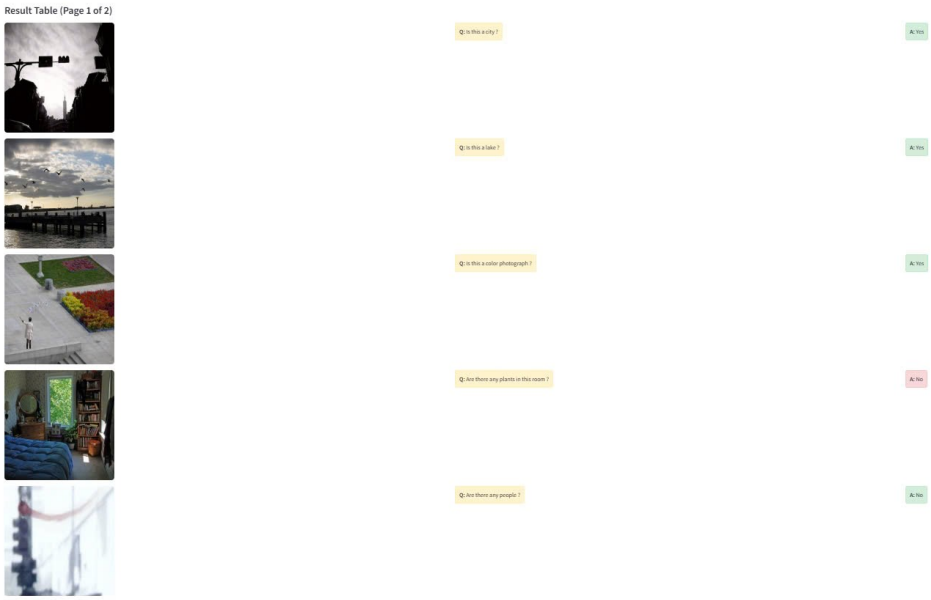
- **Word-based Tokenization**: Độ chính xác huấn luyện tăng đều từ 0.55 lên 0.85 qua 50 epoch, nhưng độ chính xác kiểm chứng dao động mạnh trong khoảng 0.50–0.54, cho thấy hiện tượng *overfitting*.
- **Subword-based Tokenization (Không dùng KD)**: Độ chính xác huấn luyện tăng từ 0.50 lên 0.90, trong khi độ chính xác kiểm chứng dao động từ 0.47 đến 0.54, cũng biểu hiện *overfitting*.

- **Subword-based Tokenization (Áp dụng KD)**: Độ chính xác huấn luyện đạt 0.90, độ chính xác kiểm chứng dao động 0.48–0.54, cải thiện nhẹ nhưng vẫn *overfit*.

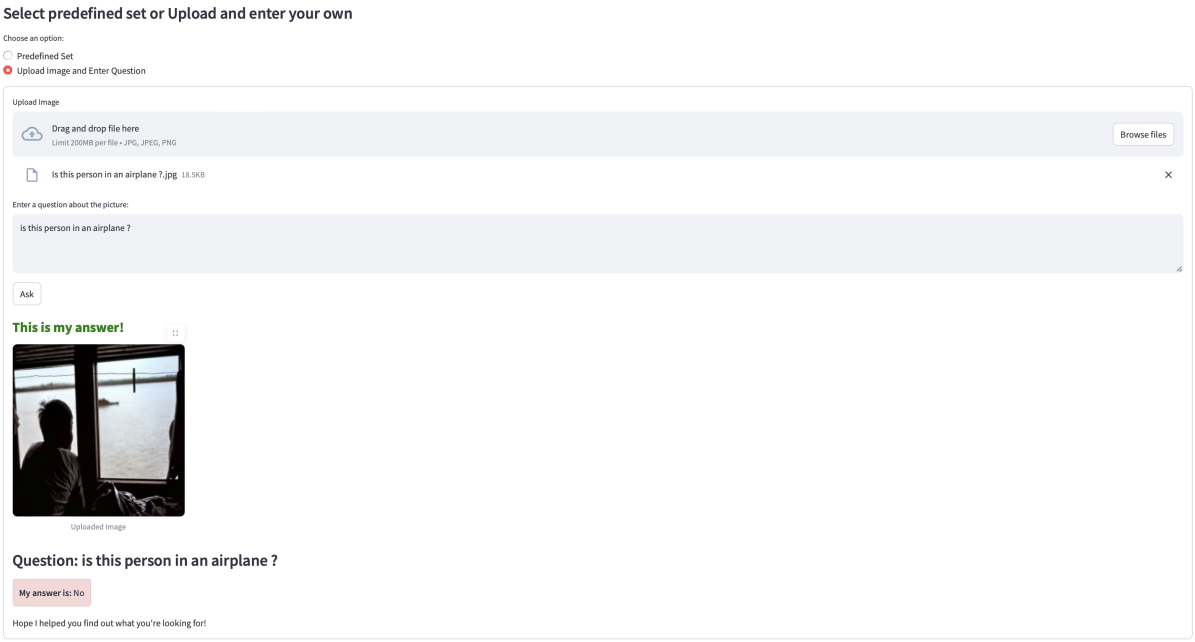
Tất cả các mô hình đều cho thấy xu hướng *overfitting* khi độ chính xác kiểm chứng không tăng tương ứng với độ chính xác huấn luyện.

3.7 Demo

Mô hình học sinh được triển khai qua Streamlit tại [3]. Kết quả demo được minh họa trong hình sau:



Hình 3.4: Dự đoán trên 5 mẫu ngẫu nhiên từ tập kiểm tra[3]



Hình 3.5: Dự đoán trên hình ảnh do người dùng tải lên[3]

3.8 Tổng kết chương

Kết chương này đã trình bày toàn diện quá trình thực nghiệm và đánh giá hệ thống VQA, từ việc thiết lập dữ liệu, xây dựng mô hình, đến triển khai ứng dụng thực tiễn. Các vấn đề đặt ra trong phần Tổng quan, như tích hợp đặc trưng hình ảnh và văn bản, tối ưu hóa mô hình thông qua KD, và đánh giá hiệu suất, đã được giải quyết thông qua việc sử dụng tập dữ liệu VQA v2.0, mô hình giáo viên (ResNet50 và Bi-LSTM), mô hình học sinh (CNN tùy chỉnh và LSTM đơn hướng), cùng với việc so sánh ba phương pháp khác nhau. Kết quả cho thấy độ chính xác huấn luyện đạt 92.44% nhưng độ chính xác xác thực chỉ dao động từ 50.82% đến 52.66%, phản ánh hiện tượng overfitting và nhu cầu cải thiện tổng quát hóa. Ứng dụng trên Streamlit minh chứng tính thực tiễn, nhưng hiệu suất vẫn cần được nâng cao. Những nhận xét này đặt nền tảng cho việc rút ra kết luận và định hướng phát triển trong chương tiếp theo – Chương 4.

4 Kết luận và hướng phát triển

4.1 Kết luận

Đề tài "Vision Question Answering" đã xây dựng thành công một hệ thống VQA cơ bản, tập trung vào trả lời câu hỏi Yes/No dựa trên hình ảnh, sử dụng tập dữ liệu VQA v2.0. Hệ thống kết hợp CNN tùy chỉnh và LSTM để trích xuất đặc trưng hình ảnh và văn bản, tích hợp BERT Tokenizer cho tiền xử lý văn bản và áp dụng phương pháp Knowledge Distillation để tối ưu hóa mô hình. Kết quả huấn luyện cho thấy độ chính xác trên tập huấn luyện đạt 92.44% sau 50 epoch, trong khi độ chính xác trên tập kiểm tra đạt 50.82%, phản ánh hiệu suất khả quan nhưng cũng chỉ ra dấu hiệu quá khớp. Việc triển khai demo trên Streamlit đã minh chứng khả năng ứng dụng thực tiễn của hệ thống.

4.2 Hướng phát triển

Để cải thiện và mở rộng đề tài, nhóm đề xuất các hướng phát triển sau:

- **Cải thiện độ chính xác:** Tăng cường kỹ thuật data augmentation, sử dụng mô hình Teacher phức tạp hơn như ViT cho xử lý hình ảnh và RoBERTa cho xử lý văn bản, đồng thời áp dụng regularization để giảm quá khớp.
- **Mở rộng loại câu hỏi:** Phát triển hệ thống hỗ trợ câu hỏi dạng văn bản tự do hoặc nhiều lựa chọn, tích hợp mô hình LLaVA để nâng cao khả năng xử lý ngữ nghĩa và hình ảnh.
- **Tối ưu hóa triển khai:** Nén mô hình Student bằng các kỹ thuật như quantization hoặc pruning, nhằm triển khai trên thiết bị di động hoặc phần cứng hạn chế.
- **Ứng dụng thực tiễn:** Tích hợp hệ thống vào các ứng dụng như trợ lý ảo, hỗ trợ người khiếm thị, hoặc công cụ giáo dục, đồng thời thử nghiệm trên dữ liệu tiếng Việt để tăng tính địa phương hóa.

Những hướng đi này sẽ giúp nâng cao hiệu suất, tính ứng dụng và đóng góp vào sự phát triển của lĩnh vực VQA tại Việt Nam.

Nhiệm vụ/ vai trò của các thành viên trong nhóm nghiên cứu

- **Trần Gia Hiển (Trưởng nhóm):** Đảm nhận vai trò nghiên cứu và áp dụng các thuật toán vào bài toán Vision Question Answering (VQA), bao gồm việc triển khai và tối ưu hóa các phương pháp như CNN, LSTM, và Knowledge Distillation để giải quyết bài toán một cách hiệu quả, đồng thời chịu trách nhiệm việc triển khai mô hình trên streamlit.
- **Đặng Ngọc Bách:** Phụ trách viết báo cáo, chuẩn bị các tài nguyên cần thiết cho dự án (bao gồm dữ liệu và môi trường tính toán), đồng thời đóng góp ý kiến và bổ sung để hoàn thiện thuật toán của nhóm, đảm bảo tính khả thi và hiệu quả.
- **Lò Văn Quyền:** Đóng góp ý kiến và đề xuất để cải thiện thuật toán của nhóm, hỗ trợ trong việc tinh chỉnh và đảm bảo tính chính xác của các phương pháp được áp dụng.

Tài liệu và Tài liệu tham khảo

[1] **Dataset:** VQA Dataset.

[2] **Mã nguồn:** Visual Question Answering (VQA) Implementation Notebook.

Phương pháp: Word-based Tokenization.

Phương pháp: Subword-based Tokenization (Không có KD) .

[3] **Website:** Demo by Streamlit.

[4] He, K., Zhang, X., Ren, S., & Sun, J. (2016). Deep Residual Learning for Image Recognition. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)* (pp. 770–778).

[5] Abhishek, A. V. S., Gurrala, V. R., & Sahoo, L. (2022). Resnet18 Model With Sequential Layer For Computing Accuracy On Image Classification Dataset. *International Journal of Creative Research Thoughts (IJCRT)*, 10(5), c176–c181.

[6] Huang, Z., Xu, W., & Yu, K. (2015). Bidirectional LSTM-CRF Models for Sequence Tagging. *arXiv preprint arXiv:1508.01991*.

[7] Hugging Face. (2023). BERT: Pre-trained Models for Natural Language Processing. *Transformers Documentation*.

[8] Vatashsky, B.-Z., & Ullman, S. (2020). VQA with No Questions-Answers Training. *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 10376–10386.

[9] Gou, J., Yu, B., Maybank, S. J., & Tao, D. (2021). Knowledge Distillation: A Survey. *International Journal of Computer Vision*, 129(6), 1789–1819.

[10] Wang, Y., Cheng, L., Duan, M., Wang, Y., Feng, Z., & Kong, S. (2023). Improving Knowledge Distillation via Regularizing Feature Norm and Direction. *arXiv preprint arXiv:2305.17007*.

[11] Miles, R., & Mikolajczyk, K. (2024). Understanding the Role of the Projector in Knowledge Distillation. *arXiv preprint arXiv:2303.11098*.

[12] Hinton, G., Vinyals, O., & Dean, J. (2015). Distilling the Knowledge in a Neural Network. *arXiv preprint arXiv:1503.02531*.

[13] Liu, H., Li, C., Wu, Q., & Lee, Y. J. (2023). Visual Instruction Tuning. *arXiv preprint arXiv:2304.08485*.

[14] Nguyen, T. T. (n.d.). Bài 6: Convolutional Neural Network. *nttuan8.com*.

[15] Dosovitskiy, A., Beyer, L., Kolesnikov, A., Weissenborn, D., Zhai, X., Unterthiner, T., Dehghani, M., Minderer, M., Heigold, G., Gelly, S., Uszkoreit, J., & Houlsby, N. (2021). An Image is Worth 16x16 Words: Transformers for Image Recognition at Scale. *Proc. Int. Conf. Learn. Represent. (ICLR)*.

[16] Liu, Y., Ott, M., Goyal, N., Du, J., Joshi, M., Chen, D., Levy, O., Lewis, M., Zettlemoyer, L., & Stoyanov, V. (2019). RoBERTa: A Robustly Optimized BERT Pretraining Approach. *arXiv preprint arXiv:1907.11692*.

[17] Kingma, D. P., & Ba, J. (2014). Adam: A Method for Stochastic Optimization. *arXiv preprint arXiv:1412.6980*.

[18] Nguyen, T. T. (n.d.). Bài 14: Long Short-Term Memory (LSTM). *nttuan8.com*.

[19] Loshchilov, I., & Hutter, F. (2016). SGDR: Stochastic Gradient Descent with Warm Restarts. *arXiv preprint arXiv:1608.03983*.

[20] Adeyemi, A., & Ojo, J. A. (2023). Automatic Classification of Cowpea Leaves Using Deep Convolutional Neural Network. *ResearchGate*.