KHOA KỸ THUẬT VÀ CÔNG NGHỆ
BỘ MÔN CÔNG NGHỆ THÔNG TIN

ĐỒ ÁN CHUYÊN NGÀNH
HỌC KỲ I, NĂM HỌC 2024 - 2025

NHẬN DẠNG ĐỐI TƯỢNG DỰA TRÊN MÔ HÌNH DENSENET

Giáo viên hướng dẫn:
ThS. Nguyễn Mộng Hiền

Sinh viên thực hiện:
Họ tên: Trần Quốc Trọng
MSSV: 110121121
Lớp: DA21TTB

Trà Vinh, tháng 01 năm 2025

KHOA KỸ THUẬT VÀ CÔNG NGHỆ
BỘ MÔN CÔNG NGHỆ THÔNG TIN

ĐỒ ÁN CHUYÊN NGÀNH
HỌC KỲ I, NĂM HỌC 2024 - 2025

NHẬN DẠNG ĐỐI TƯỢNG DỰA TRÊN MÔ HÌNH DENSENET

Giáo viên hướng dẫn:
ThS. Nguyễn Mộng Hiền

Sinh viên thực hiện:
Họ tên: Trần Quốc Trọng
MSSV: 110121121
Lớp: DA21TTB

Trà Vinh, tháng 01 năm 2025

NHẬN XÉT CỦA GIÁO VIÊN HƯỚNG DẪN

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

Trà Vinh, ngày ... tháng ... năm …..

Giáo viên hướng dẫn

(Ký tên và ghi rõ họ tên)

NHẬN XÉT CỦA THÀNH VIÊN HỘI ĐỒNG

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

Trà Vinh, ngày ... tháng ... năm …..

Thành viên hội đồng

(Ký tên và ghi rõ họ tên)

LỜI CẢM ƠN

Trước hết, tôi xin bày tỏ lòng biết ơn sâu sắc đến thầy Nguyễn Mộng Hiền, giảng viên hướng dẫn, người đã không ngừng đồng hành và tận tình chỉ bảo tôi trong suốt quá trình thực hiện đồ án này. Những chỉ dẫn chi tiết và sự hỗ trợ nhiệt tình của thầy không chỉ giúp tôi hoàn thành tốt đề tài mà còn mở rộng tầm nhìn của tôi về lĩnh vực trí tuệ nhân tạo và các ứng dụng mô hình học sâu, đặc biệt là mô hình DenseNet121.

Tôi cũng xin gửi lời cảm ơn chân thành đến Khoa Kỹ thuật và Công nghệ, Trường Đại học Trà Vinh, nơi đã trang bị cho tôi nền tảng kiến thức vững chắc và tạo điều kiện thuận lợi để tôi tiếp cận với những tài liệu khoa học có giá trị. Điều này đã giúp tôi triển khai đề tài, từ giai đoạn nghiên cứu lý thuyết cho đến thực nghiệm, một cách mạch lạc và hiệu quả.

Trong suốt quá trình thực hiện đề tài, tôi nhận thấy rằng DenseNet121 không chỉ đơn thuần là một mô hình học sâu mà còn mở ra nhiều cơ hội nghiên cứu và ứng dụng thực tiễn trong nhận diện hình ảnh. Tuy nhiên, tôi cũng hiểu rằng vẫn còn nhiều điểm cần cải thiện và hoàn thiện hơn nữa. Vì vậy, tôi rất mong nhận được những ý kiến đóng góp đến từ quý thầy cô để có thể tiếp tục nâng cao năng lực và bổ sung những kiến thức còn thiếu, đồng thời khám phá thêm các ứng dụng tiềm năng của mô hình này trong tương lai.

Tôi xin chân thành cảm ơn!

MỤC LỤC

DANH MỤC HÌNH

DANH MỤC TỪ VIẾT TẮT

STT	KÝ HIỆU VIẾT TẮT	NỘI DUNG VIẾT TẮT
1	AI	Artificial Intelligence
2	CNN	Convolutional Neural Network
3	GPU	Graphics Processing Unit
4	ReLU	Rectified Linear Unit
5	SGD	Stochastic Gradient Descent
6	TPU	Tensor Processing Unit
7	GAP	Global Average Pooling
8	BN	Batch Normalization
9	LR	Learning Rate
10	FC	Fully Connected

TÓM TẮT ĐỒ ÁN CHUYÊN NGÀNH

Đồ án chuyên ngành này tập trung vào nghiên cứu và triển khai mô hình DenseNet121 trên tập dữ liệu CIFAR-10 nhằm giải quyết bài toán phân loại hình ảnh. DenseNet121 là một trong những kiến trúc mạng nơ-ron sâu tiên tiến, nổi bật với cơ chế kết nối dày đặc giữa các tầng (Dense Connectivity), giúp giảm thiểu số lượng tham số và tối ưu hóa quá trình huấn luyện. Đồ án bao gồm các bước chính: nghiên cứu lý thuyết về DenseNet121 và tập dữ liệu CIFAR-10, thiết lập môi trường thực nghiệm trên Google Colab, huấn luyện và kiểm thử mô hình, và cuối cùng là đánh giá kết quả thông qua các phương pháp trực quan hóa.

Mô hình DenseNet121 được huấn luyện trên tập dữ liệu CIFAR-10 và đạt được độ chính xác 94.4% trên tập kiểm tra, với các chỉ số Precision, Recall và F1-score cao ở tất cả các lớp. Kết quả này thể hiện hiệu suất vượt trội của mô hình trong việc phân loại hình ảnh, cùng với đó là sự ổn định trong quá trình huấn luyện, không xuất hiện hiện tượng overfitting hay underfitting. Ngoài ra, thử nghiệm dự đoán trên dữ liệu thực tế cũng cho thấy khả năng nhận diện hiệu quả của mô hình, mở ra tiềm năng ứng dụng trong các lĩnh vực như y tế, giao thông và công nghiệp.

Mặc dù đạt được những kết quả tích cực, đồ án vẫn gặp một số hạn chế về tài nguyên tính toán và thời gian huấn luyện. Hướng phát triển trong tương lai bao gồm nghiên cứu và áp dụng các kỹ thuật tối ưu hóa tiên tiến để giảm thời gian huấn luyện, mở rộng thử nghiệm trên các tập dữ liệu lớn hơn như ImageNet, và tích hợp DenseNet121 vào các hệ thống ứng dụng thực tế để đánh giá khả năng vận hành trong điều kiện thực tiễn. Đồ án đã hoàn thành tốt các mục tiêu đề ra, đồng thời cung cấp nền tảng kiến thức và thực nghiệm vững chắc để tiếp tục phát triển và ứng dụng mô hình DenseNet121 trong tương lai.

MỞ ĐẦU

1. Lý do chọn đề tài

Trong bối cảnh công nghệ trí tuệ nhân tạo (AI) ngày càng phát triển và có ảnh hưởng mạnh mẽ đến nhiều lĩnh vực, các mô hình học sâu (Deep Learning) đóng vai trò then chốt trong việc xử lý và phân tích dữ liệu. Đặc biệt, trong lĩnh vực thị giác máy tính (Computer Vision), các mô hình mạng nơ-ron tích chập (Convolutional Neural Networks - CNNs) đã đạt được nhiều thành tựu đáng kể, giúp cải thiện hiệu suất trong các tác vụ như phân loại ảnh, nhận diện đối tượng và phát hiện bất thường.

DenseNet121, một trong những mô hình tiên tiến thuộc họ DenseNet, nổi bật nhờ khả năng tận dụng tối đa thông tin từ các tầng trong mạng bằng cách kết nối trực tiếp mọi tầng với nhau. Chính kiến trúc đặc biệt này không chỉ giúp giảm thiểu hiện tượng mất mát thông tin trong quá trình huấn luyện mà còn tăng cường khả năng học tập của mô hình trong việc xử lý các bộ dữ liệu phức tạp. Những ưu điểm này đã làm cho DenseNet121 trở thành một trong những lựa chọn hàng đầu trong các ứng dụng liên quan đến phân loại ảnh và nhận diện đối tượng.

Việc nghiên cứu và triển khai mô hình DenseNet121 trên tập dữ liệu CIFAR-10 không chỉ giúp hiểu sâu hơn về cách hoạt động và hiệu quả của mô hình mà còn mở ra tiềm năng ứng dụng rộng rãi trong thực tế, từ nhận diện y tế, giao thông, đến các ứng dụng công nghệ thông minh. Chính vì vậy, đề tài này được chọn nhằm mục tiêu không chỉ nâng cao kiến thức chuyên môn về lĩnh vực học sâu mà còn đóng góp một phần nhỏ vào việc khai thác tiềm năng của DenseNet121 trong giải quyết các vấn đề thực tiễn.

2. Mục tiêu nghiên cứu

Triển khai và đánh giá hiệu suất của mô hình DenseNet121 trên tập dữ liệu CIFAR-10 thông qua nền tảng Google Colab. Các mục tiêu cụ thể bao gồm:

- Hiểu rõ kiến trúc và cơ chế hoạt động của DenseNet121.
- Huấn luyện và kiểm thử mô hình trên tập dữ liệu CIFAR-10, đánh giá qua các chỉ số hiệu suất chính.
- Tối ưu hóa quá trình huấn luyện bằng cách sử dụng các kỹ thuật như điều chỉnh siêu tham số và tăng cường dữ liệu.
- Khai thác khả năng ứng dụng thực tiễn của mô hình trong phân loại hình ảnh.

3. Phương pháp nghiên cứu

Nghiên cứu lý thuyết: Tìm hiểu sâu về kiến trúc và nguyên lý hoạt động của DenseNet121 thông qua các tài liệu học thuật, tài nguyên trực tuyến, và mã nguồn mở.

Triển khai thực nghiệm trên Google Colab:

- Thiết lập môi trường trên Google Colab với các thư viện như PyTorch và torchvision.
- Huấn luyện mô hình DenseNet121 trên tập dữ liệu CIFAR-10 với tiền xử lý, tăng cường dữ liệu và chuẩn hóa.
- Áp dụng các chiến lược tối ưu như điều chỉnh siêu tham số, dừng sớm và giảm tốc độ học.

Đánh giá và phân tích kết quả: Nghiên cứu sử dụng các chỉ số như độ chính xác (accuracy), ma trận nhầm lẫn (confusion matrix) và báo cáo phân loại (classification report) để đánh giá hiệu suất của mô hình. Quá trình huấn luyện được trực quan hóa qua biểu đồ nhằm phân tích các yếu tố ảnh hưởng đến kết quả.

Kiểm thử và áp dụng mô hình: Mô hình được kiểm thử trên tập dữ liệu kiểm tra để đánh giá độ chính xác và khả năng khái quát hóa. Đồng thời, ứng dụng mô hình vào dự đoán các hình ảnh ngoài tập dữ liệu để kiểm tra tính ổn định và hiệu quả thực tiễn của DenseNet121.

4. Đối tượng nghiên cứu

Đối tượng nghiên cứu chính trong đề tài này là mô hình DenseNet121, một mạng nơ-ron tích chập sâu được thiết kế để tối ưu hóa việc học đặc trưng trên dữ liệu hình ảnh. Ngoài ra, tập dữ liệu CIFAR-10, bao gồm 60.000 hình ảnh thuộc 10 lớp khác nhau, được sử dụng làm nền tảng để đánh giá và kiểm chứng hiệu quả của mô hình.

5. Phạm vi nghiên cứu

Triển khai và đánh giá mô hình DenseNet121 trên tập dữ liệu CIFAR-10 thông qua nền tảng Google Colab. Các thử nghiệm sẽ giới hạn trong việc huấn luyện và kiểm thử mô hình trên tập dữ liệu này, với mục tiêu chính là phân loại hình ảnh. Đồng thời, các kỹ thuật tối ưu hóa như tăng cường dữ liệu và điều chỉnh siêu tham số cũng nằm trong phạm vi thực hiện, nhằm cải thiện hiệu suất mô hình.

CHƯƠNG 1: TỔNG QUAN

Trong thời đại công nghệ số, trí tuệ nhân tạo (AI) và học sâu (Deep Learning) đóng vai trò quan trọng trong việc giải quyết các bài toán phức tạp, đặc biệt là trong lĩnh vực thị giác máy tính. Một trong những thách thức đáng kể là xây dựng hệ thống phân loại hình ảnh hiệu quả trên các tập dữ liệu lớn và đa dạng như CIFAR-10. Tập dữ liệu này bao gồm 60.000 hình ảnh thuộc 10 lớp khác nhau, đòi hỏi mô hình phải có khả năng học đặc trưng tốt và xử lý hiệu quả để đạt được độ chính xác cao. Việc nghiên cứu và áp dụng mô hình học sâu tiên tiến, chẳng hạn như DenseNet121, không chỉ nhằm giải quyết bài toán phân loại mà còn hướng đến những ứng dụng thực tiễn trong nhiều lĩnh vực khác nhau.

Để xây dựng hệ thống đáp ứng được những yêu cầu trên, DenseNet121 được lựa chọn làm mô hình nền tảng nhờ vào thiết kế đặc biệt với cấu trúc kết nối dày đặc giữa các tầng mạng. Kiến trúc này giúp giảm thiểu hiện tượng mất mát thông tin và tối ưu hóa khả năng học đặc trưng, đồng thời cải thiện hiệu suất xử lý hình ảnh một cách đáng kể. Tập dữ liệu CIFAR-10 được sử dụng để huấn luyện và kiểm thử mô hình thông qua nền tảng Google Colab, tận dụng các công cụ mạnh mẽ như PyTorch và torchvision. Trong quá trình thực nghiệm, các kỹ thuật như tăng cường dữ liệu, chuẩn hóa và điều chỉnh siêu tham số cũng được áp dụng để đảm bảo tính hiệu quả và độ chính xác cao của hệ thống.

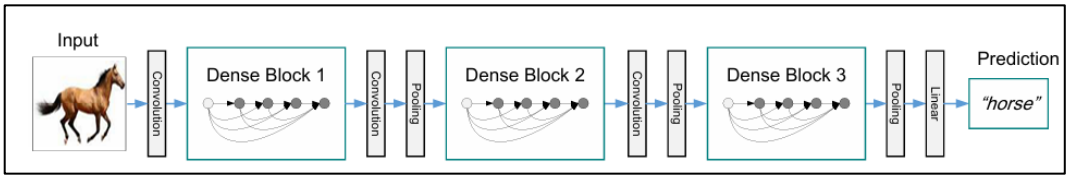
Sau quá trình triển khai và thử nghiệm, DenseNet121 đã đạt được hiệu suất phân loại hình ảnh đáng kể trên tập dữ liệu CIFAR-10 với độ chính xác cao, đáp ứng đầy đủ các tiêu chí của bài toán nghiên cứu. Mô hình này không chỉ phù hợp để giải quyết các bài toán tương tự trong lĩnh vực thị giác máy tính mà còn có thể được triển khai trong các ứng dụng thực tiễn như phân loại ảnh y tế, nhận diện vật thể trong giao thông hay các hệ thống thông minh tự động. Với thời gian xử lý được tối ưu hóa và yêu cầu tài nguyên tính toán hợp lý, hệ thống có thể dễ dàng tích hợp vào các môi trường thực tế, đảm bảo hiệu quả và tính khả thi cao.

CHƯƠNG 2: NGHIÊN CỨU LÝ THUYẾT

2.1. Tổng quan về mô hình DenseNet121

2.1.1. Lịch sử phát triển DenseNet

DenseNet (Densely Connected Convolutional Networks) là một trong những cải tiến quan trọng trong lĩnh vực học sâu, được giới thiệu lần đầu vào năm 2017 bởi Gao Huang và cộng sự trong bài báo "Densely Connected Convolutional Networks" tại hội nghị CVPR (Conference on Computer Vision and Pattern Recognition) [1]. Mô hình được phát triển nhằm giải quyết các vấn đề thường gặp trong mạng nơ-ron tích chập (CNN) truyền thống, như mất mát thông tin trong quá trình huấn luyện các mạng sâu và sự dư thừa tham số.



Hình 2. 1. Minh họa luồng xử lý dữ liệu trong mô hình DenseNet [1]

DenseNet được xây dựng dựa trên ý tưởng tái sử dụng đặc trưng từ các tầng trước đó. Điểm độc đáo của DenseNet so với các kiến trúc CNN khác là khả năng kết nối tất cả các tầng với nhau theo cách dày đặc (dense connectivity). Mỗi tầng trong DenseNet nhận đầu vào từ tất cả các tầng trước đó và truyền thông tin của nó đến tất cả các tầng sau đó. Điều này khác biệt hoàn toàn so với các mô hình trước đây, như ResNet, vốn sử dụng kết nối tắt (skip connection) để cải thiện quá trình lan truyền gradient [2].

DenseNet đã chứng minh hiệu quả vượt trội trên các tập dữ liệu lớn như ImageNet và các tập dữ liệu nhỏ hơn như CIFAR-10. Đặc biệt, mô hình DenseNet121, một biến thể tiêu chuẩn với 121 tầng, đã trở thành một lựa chọn phổ biến nhờ sự cân bằng giữa độ sâu, số lượng tham số, và hiệu suất [3].

2.1.2. Kiến trúc DenseNet121

2.1.2.1. DenseBlock và kết nối dày đặc

DenseBlock là thành phần cốt lõi của kiến trúc DenseNet, nơi mọi tầng được kết nối trực tiếp với tất cả các tầng trước đó theo kiểu kết nối dày đặc (dense connectivity). Mỗi tầng trong DenseBlock nhận đầu vào là đầu ra từ tất cả các tầng trước đó và thông tin đặc trưng đã học được, sau đó kết hợp chúng để tạo ra đầu ra mới. Điều này giúp tối ưu hóa việc tái sử dụng đặc trưng, giảm thiểu hiện tượng dư thừa thông tin, và cải thiện khả năng học tập của mô hình.

Cấu trúc của DenseBlock: Mỗi DenseBlock bao gồm một số lượng xác định các tầng, và các tầng này được sắp xếp theo trình tự.

Batch Normalization → ReLU → Convolution

Cách tổ chức này giúp giữ thông tin được chuẩn hóa, kích hoạt phi tuyến tính, và học các đặc trưng quan trọng qua phép tích chập.

Kết nối dày đặc (Dense Connectivity): Kết nối dày đặc là điểm đột phá của DenseNet, khác biệt so với các kiến trúc khác như ResNet. Trong DenseNet, các đầu ra từ mỗi tầng không chỉ được gửi đến tầng kế tiếp, mà còn được truyền đến tất cả các tầng sau đó trong cùng một khối. Điều này tạo ra một ma trận liên kết giữa các tầng, đảm bảo rằng thông tin được lưu giữ và tái sử dụng ở mọi cấp độ.

Công thức toán học biểu diễn đầu ra của tầng l trong DenseBlock:

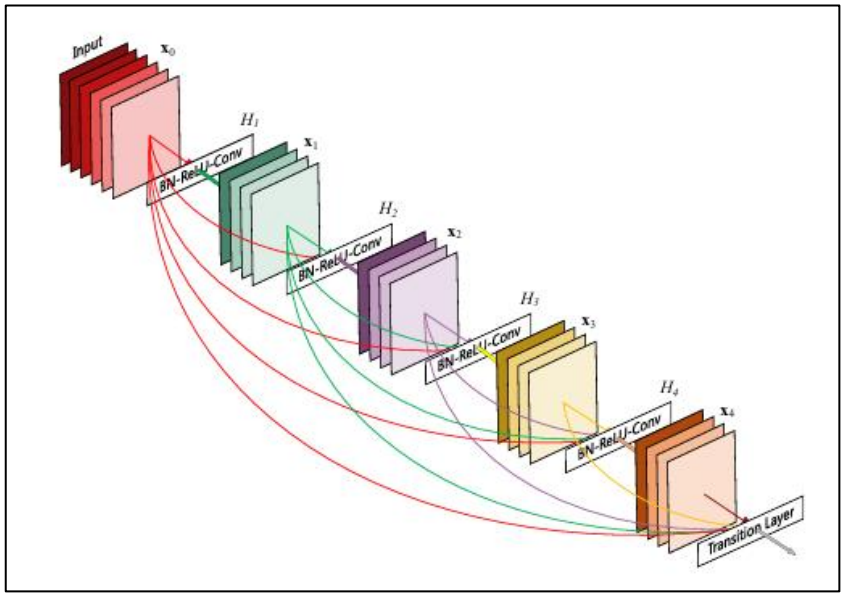
$$x_l = H_l([x_0, x_1, \dots, x_{l-1}])$$

Trong đó:

- H_l: Hàm biến đổi, thường gồm Batch Normalization, ReLU, và Convolution.
- $([x_0, x_1, \dots, x_{l-1}])$: Tập hợp đầu ra từ tất cả các tầng trước đó [1].

Ý nghĩa của DenseBlock:

- Tăng khả năng học đặc trưng sâu nhờ kết nối dày đặc.
- Giảm số lượng tham số so với các kiến trúc CNN truyền thống.
- Khắc phục hiện tượng gradient vanishing bằng cách duy trì thông tin liên tục qua các tầng.



Hình 2. 2. Minh họa cấu trúc DenseBlock và kết nối dày đặc [1]

2.1.2.2. Transition Layer

Transition Layer là thành phần quan trọng trong DenseNet, đóng vai trò kết nối giữa các DenseBlock. Nó giúp giảm kích thước dữ liệu và số lượng đặc trưng (feature maps), từ đó tối ưu hóa tài nguyên tính toán và giảm thiểu hiện tượng dư thừa.

Cấu trúc của Transition Layer:

- **Batch Normalization (BN):** Giúp ổn định quá trình huấn luyện và cải thiện tốc độ hội tụ.
- **ReLU Activation:** Kích hoạt phi tuyến tính, tạo phi tuyến tính trong mô hình.
- **Convolution (1x1):** Giảm số lượng feature maps mà không làm mất thông tin quan trọng.
- **Average Pooling (2x2):** Giảm kích thước không gian (spatial size) của feature maps, giúp làm gọn dữ liệu trước khi chuyển sang DenseBlock tiếp theo.

Vai trò của Transition Layer:

Giảm kích thước không gian: Làm nhỏ kích thước đầu ra của DenseBlock, giúp mô hình dễ dàng xử lý các dữ liệu lớn hơn.

Kiểm soát số lượng feature maps: Sử dụng tích chập 1x1 để giảm số lượng đặc trưng, làm giảm yêu cầu tài nguyên tính toán mà không ảnh hưởng đến các chất lượng đặc trưng.

Đóng vai trò cầu nối: Tạo sự liền mạch giữa các DenseBlock, đảm bảo quá trình học đặc trưng diễn ra hiệu quả.

2.1.2.3. Tầng Fully Connected

Tầng Fully Connected (FC Layer) là thành phần cuối cùng trong kiến trúc DenseNet, chịu trách nhiệm chuyển đổi đặc trưng đã học thành kết quả dự đoán. Đây là tầng tuyến tính kết nối tất cả các điểm dữ liệu đầu vào thành một đầu ra duy nhất tương ứng với các lớp phân loại.

Cấu trúc của Tầng Fully Connected:

Global Average Pooling (GAP): Thay thế các tầng Fully Connected truyền thống bằng một phương pháp gọn hơn, tính trung bình toàn bộ feature maps từ DenseBlock cuối cùng. GAP giảm thiểu số lượng tham số đáng kể so với phương pháp Fully Connected thông thường.

Fully Connected Layer: Nhận đầu vào từ GAP và sử dụng một tầng tuyến tính để phân loại dữ liệu thành các lớp.

Vai trò của Tầng Fully Connected:

- **Dự đoán kết quả:** Tầng này tạo ra xác suất cho mỗi lớp đầu ra dựa trên các đặc trưng đã học được.
- **Giảm số lượng tham số:** Sử dụng GAP thay vì các tầng Fully Connected truyền thống giúp DenseNet trở nên gọn nhẹ và hiệu quả hơn.
- **Liên kết đặc trưng với các lớp:** Mỗi lớp đầu ra được liên kết trực tiếp với các đặc trưng đã học từ toàn bộ mạng.

2.1.3. Cơ chế hoạt động

2.1.3.1. Batch Normalization và ReLU

Batch Normalization (BN): Batch Normalization là một phương pháp chuẩn hóa dữ liệu đầu vào của mỗi tầng mạng, giúp ổn định quá trình huấn luyện và tăng tốc độ hội tụ. BN được áp dụng sau mỗi phép tích chập (Convolution) và trước khi kích hoạt bằng ReLU.

Lợi ích của Batch Normalization:

- Giảm hiện tượng gradient vanishing hoặc exploding.
- Duy trì phân phối ổn định của dữ liệu, giúp mô hình học hiệu quả hơn.
- Tăng khả năng sử dụng các giá trị học (learning rate) lớn hơn.

ReLU (Rectified Linear Unit): ReLU là hàm kích hoạt phi tuyến tính, đóng vai trò loại bỏ các giá trị âm và giữ lại các giá trị dương, giúp mô hình học các đặc trưng phi tuyến một cách hiệu quả.

Công thức của ReLU:

$$f(x) = max(0, x) \quad [4]$$

Sự kết hợp giữa Batch Normalization và ReLU giúp đảm bảo rằng dữ liệu qua từng tầng luôn được chuẩn hóa, đồng thời kích hoạt mạnh mẽ các đặc trưng quan trọng.

2.1.3.2. Global Average Pooling

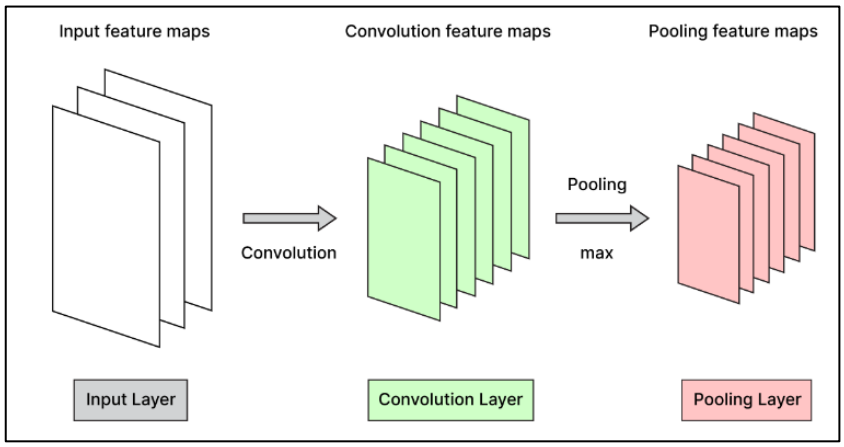
Global Average Pooling (GAP) là một cải tiến quan trọng so với các tầng Fully Connected truyền thống. GAP tính giá trị trung bình trên toàn bộ không gian của feature maps, thay vì kết nối từng neuron với từng lớp. Điều này giúp giảm đáng kể số lượng tham số mà vẫn duy trì hiệu quả học tập.

Cách hoạt động của GAP:

- Mỗi feature map được tính trung bình toàn bộ các giá trị.
- Kết quả từ GAP trở thành đầu vào của tầng phân loại cuối cùng.

Ưu điểm của GAP:

- Giảm thiểu số lượng tham số, làm cho DenseNet nhẹ hơn và nhanh hơn.
- Tránh hiện tượng overfitting thường thấy ở các tầng Fully Connected truyền thống.
- Đảm bảo mỗi feature map được liên kết trực tiếp với một lớp đầu ra, tăng cường khả năng tổng quát hóa.



Hình 2. 3. Minh họa tầng Global Average Pooling [8]

2.1.3.3. Tái sử dụng đặc trưng

Một trong những cơ chế cốt lõi của DenseNet là việc tái sử dụng đặc trưng giữa các tầng, thông qua kết nối dày đặc (dense connectivity) trong DenseBlock. Mỗi tầng không chỉ nhận đầu vào từ tầng trước mà còn từ tất cả các tầng trước đó trong cùng một DenseBlock. Điều này tạo ra một luồng thông tin liên tục và tối ưu hóa và học đặc trưng.

Ý nghĩa của tái sử dụng đặc trưng:

- **Tăng hiệu quả học đặc trưng:** Mỗi tầng có khả năng học các đặc trưng từ đầu vào gốc và từ tất cả các tầng trước.
- **Giảm số lượng tham số:** Nhờ việc tái sử dụng, DenseNet có số lượng tham số thấp hơn đáng kể so với các kiến trúc CNN truyền thống.
- **Khắc phục vấn đề gradient vanishing:** Do các kết nối trực tiếp, gradient có thể lan truyền ngược một cách dễ dàng đến các tầng trước đó.

2.1.4. Ưu điểm và hạn chế

Ưu điểm:

- **Tái sử dụng đặc trưng hiệu quả:** DenseNet tận dụng cơ chế kết nối dày đặc, giúp tái sử dụng đặc trưng giữa các tầng, từ đó giảm số lượng tham số và tăng hiệu quả học đặc trưng.

- **Giảm hiện tượng gradient vanishing:** Nhờ kết nối trực tiếp giữa các tầng, thông tin gradient dễ dàng lan truyền ngược qua các tầng sâu, giúp mô hình dễ huấn luyện hơn.
- **Tối ưu hóa tài nguyên tính toán:** Với số lượng tham số ít hơn so với các mô hình như ResNet hay VGG, DenseNet vẫn đạt được hiệu suất cao trên nhiều tập dữ liệu.
- **Tăng khả năng tổng quát hóa:** DenseNet có khả năng học các đặc trưng sâu, dẫn đến hiệu suất tốt hơn trong các bài toán thị giác máy tính như phân loại và nhận diện hình ảnh.
- **Khả năng mở rộng tốt:** DenseNet dễ dàng được triển khai và mở rộng trên các tập dữ liệu lớn nhờ kiến trúc tối ưu và hiệu quả.

Hạn chế:

- **Yêu cầu tài nguyên cao:** Dù số lượng tham số thấp, DenseNet vẫn cần lượng lớn tài nguyên GPU để xử lý các kết nối dày đặc giữa các tầng.
- **Thời gian huấn luyện lâu:** Việc tính toán các kết nối dày đặc đòi hỏi thời gian nhiều hơn so với các mô hình như ResNet.
- **Kém hiệu quả trên dữ liệu lớn và phức tạp:** DenseNet có thể gặp khó khăn khi áp dụng trên các tập dữ liệu cực lớn hoặc phức tạp mà không có sự điều chỉnh phù hợp.

2.1.5. So sánh với các mô hình khác

2.1.5.1. So sánh với ResNet

Điểm tương đồng: Cả DenseNet và ResNet đều giải quyết vấn đề gradient vanishing thông qua các kết nối đặc biệt. ResNet sử dụng kết nối tắt (skip connection), trong khi DenseNet sử dụng kết nối dày đặc.

Điểm khác biệt:

- **Cách kết nối:** ResNet chỉ kết nối đầu vào với đầu ra của tầng kế tiếp, còn DenseNet kết nối mọi tầng với nhau trong cùng một DenseBlock.
- **Số lượng tham số:** DenseNet yêu cầu ít tham số hơn nhờ cơ chế tái sử dụng đặc trưng.
- **Hiệu suất:** DenseNet thường có hiệu suất cao hơn trên các tập dữ liệu nhỏ như CIFAR-10, trong khi ResNet hoạt động tốt trên các tập dữ liệu lớn hơn như ImageNet.

Kết luận: DenseNet vượt trội hơn trong việc tối ưu hóa tài nguyên và đạt hiệu suất cao trên dữ liệu nhỏ, trong khi ResNet phù hợp hơn với các bài toán yêu cầu mạng sâu hơn.

2.1.5.2. So sánh với VGG và AlexNet

So với VGG:

- **Tham số:** DenseNet yêu cầu ít tham số hơn so với VGG, nhờ việc tái sử dụng đặc trưng giữa các tầng.
- **Hiệu quả:** DenseNet có hiệu suất vượt trội hơn trong việc học đặc trưng sâu, trong khi VGG có xu hướng dư thừa các tham số với các tầng tích chập sâu và Fully Connected lớn.

So với AlexNet:

- **Hiện đại hơn:** DenseNet được thiết kế tối ưu và hiện đại hơn so với AlexNet, vốn là một trong những mô hình CNN đầu tiên.
- **Hiệu suất:** DenseNet dễ dàng vượt qua AlexNet trong hầu hết các bài toán phân loại hình ảnh.

2.1.6. Ứng dụng thực tiễn của DenseNet121

DenseNet121 có ứng dụng rộng rãi trong các lĩnh vực thực tiễn nhờ hiệu suất cao và khả năng học sâu vượt trội. Một số ứng dụng tiêu biểu bao gồm:

Y tế:

- Nhận diện hình ảnh X-quang và MRI để phát hiện bệnh.
- Phân loại hình ảnh tế bào trong nghiên cứu y sinh học.

Giao thông:

- Nhận diện biển báo giao thông trong các hệ thống xe tự lái.
- Phân loại và theo dõi đối tượng trong môi trường giao thông phức tạp.

Hệ thống giám sát thông minh:

- Nhận diện khuôn mặt và phân tích hành vi trong các hệ thống an ninh.
- Giám sát công nghiệp để phát hiện lỗi hoặc bất thường trong dây chuyền sản xuất.

Thị giác máy tính trong sản xuất:

- Phân loại sản phẩm trong các dây chuyền sản xuất tự động.
- Kiểm tra chất lượng hình ảnh trong các ngành công nghiệp sản xuất các hình ảnh số.

DenseNet121 không chỉ là một công cụ mạnh mẽ trong học thuật mà còn được ứng dụng rộng rãi trong thực tế, tạo ra giá trị đáng kể trong nhiều ngành công nghiệp.

2.2. Tổng quan về tập dữ liệu CIFAR-10

2.2.1. Giới thiệu về CIFAR-10

2.2.1.1. Nguồn gốc tập dữ liệu CIFAR-10

CIFAR-10 (Canadian Institute for Advanced Research) là một trong những tập dữ liệu tiêu chuẩn được sử dụng rộng rãi trong lĩnh vực học sâu và thị giác máy tính. Tập dữ liệu này được xây dựng bởi Alex Krizhevsky, Geoffrey Hinton, và Vinod Nair vào năm 2009 với mục tiêu tạo ra một tập dữ liệu nhỏ gọn nhưng đủ đa dạng, nhằm đánh giá hiệu suất của các mô hình học sâu trên bài toán phân loại hình ảnh [5].

CIFAR-10 được phát triển dựa trên tập dữ liệu Tiny Images, bao gồm hơn 80 triệu hình ảnh nhỏ. Trong đó, nhóm nghiên cứu đã chọn lọc 60.000 hình ảnh có kích thước 32x32 điểm ảnh, đại diện cho 10 lớp đối tượng khác nhau như máy bay, ô tô, chim, mèo, và nhiều hơn nữa[5].

Mục tiêu chính khi xây dựng tập dữ liệu CIFAR-10 là cung cấp một bộ dữ liệu tiêu chuẩn, dễ sử dụng và dễ tích hợp cho các nghiên cứu học sâu. Cụ thể:

- **Làm tiêu chuẩn:** CIFAR-10 được sử dụng như một thước đo hiệu suất của các mô hình phân loại hình ảnh.
- **Đánh giá khả năng học đặc trưng:** Nhờ tính đa dạng của dữ liệu, CIFAR-10 cho phép kiểm tra khả năng học đặc trưng của các mô hình, đặc biệt là các mô hình sâu [5].
- **Khuyến khích nghiên cứu cải tiến:** CIFAR-10 tạo cơ hội cho các nhà nghiên cứu thử nghiệm các cải tiến mới, giúp tối ưu hóa các thuật toán và mô hình học sâu [6].

Hình 2. 4. Minh họa bộ dữ liệu CIFAR-10 [6]

2.2.1.2. Cấu trúc tập dữ liệu CIFAR-10

Tập dữ liệu CIFAR-10 được thiết kế với cấu trúc đơn giản, rõ ràng và dễ sử dụng, nhằm phục vụ các nghiên cứu và thử nghiệm trong lĩnh vực học sâu. Với tổng cộng 60.000 hình ảnh, CIFAR-10 được phân chia rõ ràng thành tập huấn luyện (training set) và tập kiểm tra (test set), cùng với các đặc điểm kỹ thuật cụ thể. Và phân chia dữ liệu như sau:

- **Tập huấn luyện:** 50.000 hình ảnh, chiếm 83.33% tổng dữ liệu, được sử dụng để huấn luyện các mô hình học sâu.
- **Tập kiểm tra:** 10.000 hình ảnh còn lại, chiếm 16.67% trên tổng dữ liệu, được sử dụng để đánh giá hiệu suất của mô hình trên dữ liệu chưa học trước đó [6].

Mỗi hình ảnh trong CIFAR-10 có kích thước cố định là 32x32 điểm ảnh, tương ứng với ma trận 3 chiều gồm:

- **Chiều cao (Height):** 32 điểm ảnh.
- **Chiều rộng (Width):** 32 điểm ảnh.

Kênh màu gồm 3 kênh RGB:

- Red
- Green
- Blue

Dữ liệu mỗi kênh được biểu diễn dưới dạng số nguyên 8-bit (giá trị từ 0 đến 255). Ngoài ra, CIFAR-10 được lưu trữ dưới dạng nhị phân (binary), mỗi file chứa dữ liệu hình ảnh và nhãn tương ứng. Trong các thư viện như PyTorch và TensorFlow, dữ liệu này thường được nạp dưới dạng Tensor để thuận tiện cho việc xử lý và huấn luyện [7].

Đồng thời, bộ dữ liệu CIFAR-10 được phân thành 10 lớp, mỗi lớp gồm 6.000 hình ảnh. Các lớp này không trùng lặp và được lựa chọn để đại diện cho các đối tượng phổ biến trong thực tế [6]. Dưới đây là danh sách các lớp tương ứng:

- Máy bay (Airplane)
- Ô tô (Automobile)
- Chim (Bird)
- Mèo (Cat)
- Hươu (Deer)
- Chó (Dog)
- Ếch (Frog)

- Ngựa (Horse)
- Tàu thủy (Ship)
- Xe tải (Truck)

Đặc điểm nổi bật của phân loại lớp:

- **Số lượng cân bằng:** Mỗi lớp có 6.000 hình ảnh.
- **Tính đa dạng:** Các hình ảnh trong mỗi lớp có sự khác biệt rõ ràng về góc nhìn, ánh sáng, và bối cảnh, tạo ra thách thức cho các mô hình phân loại [6].

2.2.2. Đặc điểm nổi bật

Tập dữ liệu CIFAR-10 sở hữu nhiều đặc điểm nổi bật giúp nó trở thành một trong những bộ dữ liệu tiêu chuẩn (benchmark) được sử dụng rộng rãi trong lĩnh vực học sâu và thị giác máy tính. Dưới đây là các đặc điểm chính:

Kích thước nhỏ gọn nhưng đủ phức tạp: CIFAR-10 được thiết kế với kích thước hình ảnh 32x32 điểm ảnh, nhỏ gọn và phù hợp để thử nghiệm trên các hệ thống với tài nguyên tính toán hạn chế [6].

Sự đa dạng trong dữ liệu: Các hình ảnh trong CIFAR-10 thể hiện sự phong phú về góc nhìn, ánh sáng, và môi trường, đại diện cho nhiều tình huống thực tế khác nhau. Mỗi lớp có sự khác biệt rõ ràng về đặc trưng hình ảnh, giúp các mô hình phải học sâu để phân biệt [6].

Phân bổ dữ liệu cân bằng: CIFAR-10 được xây dựng với số lượng hình ảnh đồng đều giữa các lớp, mỗi lớp có 6.000 hình ảnh. Điều này giúp đảm bảo rằng không có lớp nào bị thiên lệch, tạo điều kiện lý tưởng để đánh giá công bằng hiệu suất của mô hình [7].

Khả năng tích hợp dễ dàng: CIFAR-10 có sẵn trên nhiều thư viện học sâu như PyTorch, TensorFlow, và Keras, với các phương pháp tải dữ liệu đơn giản, dễ sử dụng. Điều này giúp nhà nghiên cứu nhanh chóng tích hợp dữ liệu vào các mô hình của họ mà không cần xử lý thủ công [7].

Tính ứng dụng rộng rãi: Tiêu chuẩn đánh giá (Benchmark): CIFAR-10 thường được sử dụng để kiểm tra hiệu suất của các mô hình học sâu từ đơn giản (như MLP, CNN cơ bản) đến các mô hình tiên tiến (ResNet, DenseNet).

Thử nghiệm các cải tiến mới: Nhờ sự đa dạng và cân bằng của dữ liệu, CIFAR-10 trở thành môi trường lý tưởng để kiểm tra các cải tiến mới về kiến trúc mạng, thuật toán tối ưu hóa, và tiền xử lý dữ liệu [6].

2.2.3. Vai trò của CIFAR-10 trong thị giác máy tính

Tập dữ liệu CIFAR-10 đóng vai trò quan trọng trong lĩnh vực thị giác máy tính, không chỉ nhờ sự phổ biến mà còn bởi khả năng hỗ trợ các nghiên cứu và ứng dụng thực tế. Dưới đây là các vai trò nổi bật của CIFAR-10 trong lĩnh vực này:

Tiêu chuẩn đánh giá mô hình (Benchmark): CIFAR-10 là tập dữ liệu tiêu chuẩn được sử dụng để đánh giá hiệu suất của các mô hình học sâu, từ CNN cơ bản đến các kiến trúc tiên tiến như ResNet và DenseNet [6]. Với kích thước nhỏ gọn và dữ liệu cân bằng, tập dữ liệu này tạo điều kiện lý tưởng cho việc kiểm thử và so sánh mô hình.

Thử nghiệm các cải tiến mới: CIFAR-10 hỗ trợ thử nghiệm các ý tưởng mới như kiến trúc mạng, thuật toán tối ưu hóa (dừng sớm, giảm tốc độ học), và các kỹ thuật tăng cường dữ liệu (Data Augmentation). Việc xử lý dữ liệu nhỏ gọn giúp giảm thời gian và tài nguyên tính toán [7].

Đào tạo và kiểm thử: CIFAR-10 cung cấp 50.000 hình ảnh huấn luyện và 10.000 hình ảnh kiểm tra, đảm bảo sự đa dạng trong dữ liệu. Điều này giúp các mô hình học sâu phân biệt hiệu quả giữa các lớp đối tượng, từ mèo, chó đến xe tải và ô tô [6].

Ứng dụng trong giáo dục và thực tiễn:

- **Giáo dục:** Thường xuyên được sử dụng trong các khóa học về học sâu để đào tạo và thực hành mô hình.
- **Xã hội**: Là cơ sở để phát triển các hệ thống nhận diện hình ảnh trong y tế, giao thông, và công nghiệp.

2.3. Các kỹ thuật tiền xử lý và tối ưu hóa

2.3.1. Tiền xử lý dữ liệu

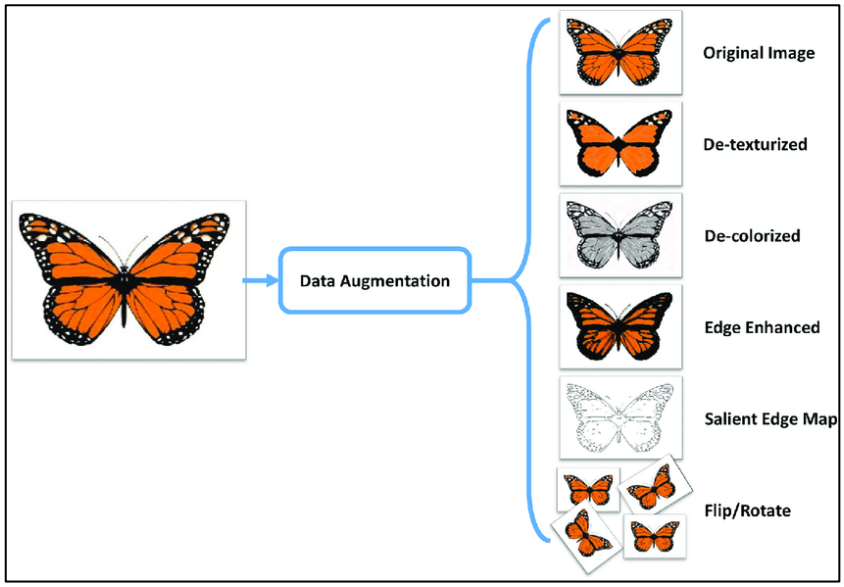
2.3.1.1. Tăng cường dữ liệu

Tăng cường dữ liệu (Data Augmentation) là một trong những kỹ thuật quan trọng trong học sâu, đặc biệt là khi làm việc với các tập dữ liệu nhỏ như CIFAR-10. Phương pháp này giúp mở rộng tập dữ liệu và tạo ra các biến thể khác nhau từ dữ liệu gốc, làm tăng tính đa dạng và giảm hiện tượng overfitting trong quá trình huấn luyện [9].

Các kỹ thuật tăng cường dữ liệu phổ biến trên CIFAR-10:

- **Random Crop:** Cắt ngẫu nhiên một phần nhỏ của hình ảnh và sau đó điều chỉnh kích thước lại về 32x32.
- **Random Horizontal Flip:** Lật ngẫu nhiên hình ảnh theo chiều ngang, làm tăng khả năng tổng quát hóa của mô hình trong việc nhận diện các đối tượng.

- **Random Rotation:** Xoay hình ảnh trong một phạm vi nhỏ, tạo ra các biến thể hình ảnh mới với góc nhìn khác nhau.
- **Color Jitter:** Điều chỉnh độ sáng, độ tương phản, độ bão hòa, và sắc độ của hình ảnh, giúp mô hình thích nghi với các điều kiện ánh sáng khác nhau trong thực tế [10].



Hình 2. 5. Minh họa phương pháp tăng cường dữ liệu [9]

Ý nghĩa của tăng cường dữ liệu:

- Tăng số lượng mẫu dữ liệu để giảm overfitting.
- Giúp mô hình học cách nhận diện đối tượng từ nhiều biến thể hình ảnh.
- Cải thiện hiệu suất mô hình trên dữ liệu kiểm tra [9].

2.3.1.2. Chuẩn hóa dữ liệu

Chuẩn hóa dữ liệu là một bước không thể thiếu trong quá trình xử lý dữ liệu trước khi đưa vào mô hình học sâu. Trên CIFAR-10, hình ảnh RGB có giá trị điểm ảnh từ 0 đến 255. Chuẩn hóa giúp chuyển đổi giá trị này về một phạm vi cố định (thường là [-1, 1] hoặc [0, 1]), giúp mô hình học hiệu quả hơn [10].

Quy trình chuẩn hóa trên CIFAR-10:

- Tính giá trị trung bình (Mean) và độ lệch chuẩn (Std) của các kênh R, G, B trong toàn bộ tập dữ liệu.
- Công thức chuẩn hóa từng điểm ảnh:

$$x' = \frac{x - \mu}{\sigma} \quad [11]$$

Trong đó:

- **x:** Giá trị điểm ảnh gốc.
- μ**:** Giá trị trung bình của kênh màu.
- σ**:** Độ lệch chuẩn của kênh màu.

Ý nghĩa của chuẩn hóa:

- Giảm chênh lệch về phân phối dữ liệu giữa các kênh màu, giúp mô hình hội tụ nhanh hơn trong quá trình huấn luyện.
- Cải thiện độ ổn định của gradient trong quá trình lan truyền ngược [11].

2.3.2. Phương pháp tối ưu hóa

2.3.2.1. Điều chỉnh siêu tham số

Điều chỉnh siêu tham số là quá trình tìm kiếm các giá trị tối ưu cho các tham số như learning rate, batch size, và weight decay. Trên CIFAR-10, việc điều chỉnh siêu tham số đóng vai trò quan trọng trong việc cải thiện hiệu suất mô hình [12].

Các siêu tham số cần tối ưu:

- **Learning Rate (LR):** Quyết định tốc độ hội tụ của mô hình.
  - LR quá lớn có thể làm mất ổn định mô hình.
  - LR quá nhỏ sẽ khiến mô hình học chậm.
- **Batch Size:** Số lượng mẫu trong mỗi batch. Batch size nhỏ giúp mô hình cập nhật nhanh hơn nhưng tăng dao động, trong khi batch size lớn làm giảm dao động nhưng cần tài nguyên cao hơn.
- **Weight Decay:** Một kỹ thuật điều chỉnh trọng số để giảm thiểu overfitting bằng cách thêm một thuật toán phạt (regularization term) vào hàm mất mát, giúp kiểm soát độ lớn của các trọng số trong mô hình.

Kỹ thuật tối ưu hóa:

Grid Search: Dò tìm giá trị tốt nhất bằng cách kiểm tra tất cả các tổ hợp tham số trong một lưới giá trị.

Random Search: Chọn ngẫu nhiên các giá trị tham số trong phạm vi xác định để giảm thời gian tìm kiếm [12].

2.3.2.2. Dừng sớm

Dừng sớm là một kỹ thuật ngăn chặn overfitting bằng cách ngừng huấn luyện khi hiệu suất trên tập kiểm tra không cải thiện sau một số lượng epoch nhất định [13].

Quy trình:

- Theo dõi giá trị loss hoặc accuracy trên tập kiểm tra trong mỗi epoch.
- Ngừng huấn luyện nếu không có cải thiện đáng kể sau một số epoch liên tiếp.

Ý nghĩa:

- Ngăn chặn mô hình học quá mức (overfitting).
- Tiết kiệm thời gian và tài nguyên tính toán.

2.3.2.3. Giảm tốc độ học

Learning Rate Scheduler là một phương pháp giảm dần learning rate trong quá trình huấn luyện để cải thiện hiệu suất hội tụ của mô hình [13].

Các chiến lược phổ biến:

- **Step Decay:** Giảm learning rate theo từng giai đoạn (epoch).
- **Exponential Decay:** Giảm learning rate theo hàm mũ.
- **Cosine Annealing:** Giảm learning rate theo đường cong hình sin để đạt hiệu quả cao hơn [13].

2.3.2.4. Tối ưu hóa bằng Stochastic Gradient Descent

SGD (Stochastic Gradient Descent) là một trong những thuật toán tối ưu hóa quan trọng và phổ biến nhất trong học sâu. Đây là phương pháp được sử dụng rộng rãi để cập nhật trọng số trong quá trình huấn luyện các mô hình, giúp giảm hàm mất mát (loss function) một cách hiệu quả.

Ngoài ra, SGD là một biến thể của thuật toán Gradient Descent thông thường, trong đó:

- **Gradient Descent truyền thống:** Cập nhật trọng số dựa trên toàn bộ dữ liệu (batch size = toàn bộ dataset).
- **Stochastic Gradient Descent:** Cập nhật trọng số dựa trên một mẫu dữ liệu ngẫu nhiên hoặc một batch nhỏ, làm giảm nhu cầu tính toán và tăng tốc độ huấn luyện.

Công thức cập nhật trọng số trong SGD:

$$w = w - \eta . \nabla L_i(w) \quad [18]$$

Trong đó:

- w : Trọng số (parameters) của mô hình.
- η : Learning rate, xác định tốc độ cập nhật.

- $\nabla L_i(w)$: Gradient của hàm mất mát L_i tính trên một mẫu dữ liệu hoặc batch nhỏ i [18].

2.4. Tổng quan về Google Colab

2.4.1. Giới thiệu Google Colab

Google Colab (Collaboratory) là một nền tảng trực tuyến miễn phí do Google phát triển, cho phép người dùng viết và thực thi mã Python trực tiếp trên trình duyệt web. Đây là công cụ được thiết kế đặc biệt để hỗ trợ các nghiên cứu về học máy (Machine Learning) và học sâu (Deep Learning) mà không yêu cầu cấu hình phần cứng phức tạp [14].

Hình 2. 6. Minh họa Google Colab [14]

Đặc điểm chính:

- Cung cấp tài nguyên GPU và TPU miễn phí.
- Hỗ trợ mạnh mẽ các thư viện học sâu như:
  - TensorFlow.
  - PyTorch.
  - Keras.
- Dễ dàng chia sẻ và lưu trữ kết quả trên Google Drive hoặc GitHub.

Google Colab đã trở thành một công cụ tiêu chuẩn cho các nhà nghiên cứu, lập trình viên, và sinh viên, đặc biệt trong bối cảnh phát triển nhanh chóng của trí tuệ nhân tạo và trong lĩnh vực nghiên cứu, học máy.

2.4.2. Tính năng nổi bật

Hỗ trợ GPU và TPU miễn phí: Google Colab cung cấp tài nguyên GPU và TPU miễn phí, giúp tăng tốc đáng kể các tác vụ huấn luyện và kiểm thử mô hình học sâu. Điều này hữu ích đối với các bài toán phức tạp đòi hỏi thời gian tính toán lớn [15].

Môi trường Python tích hợp sẵn: Với Google Colab, người dùng có thể truy cập ngay các thư viện Python phổ biến như NumPy, Pandas, Matplotlib, TensorFlow, và PyTorch mà không cần thực hiện bất kỳ bước cài đặt thủ công nào.

Tự động lưu trữ và chia sẻ: Colab cho phép lưu trữ các notebook trực tiếp trên Google Drive, đảm bảo dữ liệu luôn được an toàn và dễ dàng truy cập. Ngoài ra, các dự án có thể được chia sẻ nhanh chóng với đồng nghiệp hoặc bạn học thông qua một liên kết đơn giản.

Tích hợp liền mạch với Google Drive và GitHub: Người dùng có thể dễ dàng tải dữ liệu từ Google Drive hoặc đồng bộ hóa các dự án với GitHub. Điều này giúp quản lý tệp dữ liệu và mã nguồn trở nên thuận tiện hơn, đặc biệt trong các dự án lớn yêu cầu tính hợp tác cao.

Hỗ trợ trực quan hóa mạnh mẽ: Google Colab tích hợp đầy đủ các thư viện trực quan hóa dữ liệu như Matplotlib và Seaborn, cho phép người dùng trình bày kết quả dưới dạng biểu đồ và hình ảnh trực quan. Điều này rất hữu ích trong việc phân tích và báo cáo kết quả nghiên cứu.

2.4.1. Lợi ích và hạn chế

2.4.1.1. Lợi ích

Miễn phí và dễ tiếp cận: Google Colab cung cấp GPU và TPU miễn phí, giúp giảm thiểu chi phí phần cứng cho các nhà nghiên cứu và sinh viên.

Dễ sử dụng và tích hợp: Môi trường trực quan, không yêu cầu cài đặt phức tạp, và tích hợp với các công cụ lưu trữ như Google Drive và GitHub.

Tăng tốc nghiên cứu: Tài nguyên GPU và TPU giúp rút ngắn thời gian huấn luyện các mô hình lớn như DenseNet121.

Hỗ trợ cộng đồng lớn: Colab được sử dụng rộng rãi và có nhiều tài liệu hướng dẫn, giúp giải quyết nhanh các vấn đề khi sử dụng [15].

2.4.1.2. Hạn chế

Giới hạn tài nguyên: GPU/TPU miễn phí nhưng có giới hạn thời gian sử dụng khoảng 4 giờ 10 phút cho GPU và 2 giờ 30 phút cho TPU trên mỗi phiên làm việc.

Phụ thuộc vào internet: Colab yêu cầu kết nối internet liên tục, dễ gây gián đoạn khi mạng không ổn định.

Không kiểm soát hoàn toàn tài nguyên: Người dùng không thể lựa chọn loại GPU/TPU cụ thể hoặc mở rộng tài nguyên khi cần.

Bảo mật dữ liệu: Lưu trữ dữ liệu trên nền tảng đám mây tiềm ẩn rủi ro bảo mật, đặc biệt khi làm việc với các dữ liệu nhạy cảm [16].

CHƯƠNG 3: HIỆN THỰC HÓA NGHIÊN CỨU

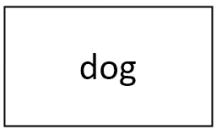
3.1. Mô tả bài toán

Bài toán được thực hiện nhằm giải quyết vấn đề phân loại hình ảnh thuộc tập dữ liệu CIFAR-10 bằng mô hình DenseNet121. Mục tiêu chính của bài toán là xây dựng một mô hình học sâu có khả năng nhận diện và phân loại chính xác các đối tượng từ ảnh, thuộc 10 lớp khác nhau như máy bay, ô tô, chim, mèo, và chó.

Hình ảnh đầu vào của bài toán là các ảnh màu RGB kích thước 32×32 điểm ảnh, thuộc tập dữ liệu CIFAR-10 tương tự như hình 3.1 dưới đây:

Hình 3. 1. Minh họa hình ảnh đầu vào (Input)

Sau khi qua các bước tiền xử lý và huấn luyện, mô hình sẽ dự đoán lớp của từng hình ảnh đầu vào và đưa ra kết quả đầu ra là một nhãn tương ứng như ảnh 3.2 như sau:



Hình 3. 2. Minh họa kết quả đầu ra (Output) là một nhãn dự đoán

3.2. Chuẩn bị dữ liệu

3.2.1. Tải dữ liệu và tiền xử lý

Quá trình chuẩn bị dữ liệu là bước quan trọng nhằm đảm bảo dữ liệu đầu vào có chất lượng tốt và phù hợp để mô hình DenseNet121 học và phân tích hiệu quả. Tập dữ liệu được sử dụng trong nghiên cứu này là CIFAR-10, một bộ dữ liệu phổ biến trong lĩnh vực thị giác máy tính. Các bước thực hiện bao gồm:

1. Tải dữ liệu CIFAR-10:

- CIFAR-10 được tải từ thư viện Torchvision, một phần của PyTorch.

- Bộ dữ liệu bao gồm 60.000 hình ảnh màu, kích thước 32×32 điểm ảnh, thuộc 10 lớp khác nhau, mỗi lớp có 6.000 hình ảnh.

2. Tập dữ liệu được chia thành:

- **Tập huấn luyện (Training Set):** Gồm 50.000 hình ảnh, được sử dụng để tối ưu hóa các trọng số trong quá trình học.
- **Tập kiểm tra (Test Set):** Gồm 10.000 hình ảnh, dùng để đánh giá hiệu suất của mô hình.
- Nếu dữ liệu chưa tồn tại trong thư mục chỉ định, chương trình sẽ tự động tải về và lưu trữ cục bộ.

3. Tiền xử lý dữ liệu:

Để tăng tính hiệu quả và khả năng học của mô hình, dữ liệu được xử lý thông qua các bước sau:

Tăng cường dữ liệu (Data Augmentation):

- **Random Crop:** Cắt ngẫu nhiên các vùng ảnh nhỏ từ hình gốc, sau đó điều chỉnh kích thước về 32×32.
- **Random Horizontal Flip:** Lật ngẫu nhiên hình ảnh theo chiều ngang để tăng tính đa dạng.
- **Random Rotation:** Xoay hình ảnh trong một phạm vi nhất định để mô hình học được nhiều góc nhìn.

Các kỹ thuật này giúp mở rộng dữ liệu, giảm thiểu hiện tượng overfitting và cải thiện khả năng khái quát hóa của mô hình.

Chuẩn hóa dữ liệu (Normalization):

- Áp dụng chuẩn hóa dựa trên giá trị trung bình (mean) và độ lệch chuẩn (std) của từng kênh màu RGB.
- Việc chuẩn hóa giúp giảm sự chênh lệch về phân phối dữ liệu giữa các kênh màu, từ đó tăng tốc độ hội tụ và ổn định trong quá trình huấn luyện.

Tách dữ liệu:

Tập dữ liệu được tách thành các phần huấn luyện và kiểm tra. Mỗi phần được tổ chức thành các batch để sử dụng hiệu quả trong quá trình huấn luyện và đánh giá.

- **Tập huấn luyện:** Được áp dụng cả tăng cường dữ liệu và chuẩn hóa.
- **Tập kiểm tra:** Chỉ áp dụng chuẩn hóa, giữ nguyên cấu trúc dữ liệu ban đầu.

4. Tổ chức dữ liệu và tăng tốc xử lý:

- Dữ liệu được chia thành các batch nhỏ để mô hình có thể xử lý từng phần thay vì toàn bộ dữ liệu một lúc. Việc này giúp tối ưu hóa bộ nhớ và tăng hiệu suất huấn luyện.
- Quá trình tải và tiền xử lý dữ liệu được thực hiện với multiprocessing, giúp giảm thời gian chờ và tối ưu hóa việc sử dụng tài nguyên hệ thống.

Nhờ quá trình tải và tiền xử lý được tổ chức cẩn thận, dữ liệu đảm bảo chất lượng cao, đáp ứng đầy đủ yêu cầu để huấn luyện mô hình DenseNet121 một cách hiệu quả..

3.2.2. Phân tích dữ liệu

Trong quá trình huấn luyện mô hình học sâu, việc phân tích và hiểu rõ cấu trúc dữ liệu đầu vào đóng vai trò then chốt để đảm bảo chất lượng và độ tin cậy của mô hình. Dưới đây là các bước phân tích dữ liệu CIFAR-10 được thực hiện để đánh giá sự phân bố và chất lượng dữ liệu:

1. Phân phối dữ liệu huấn luyện và kiểm tra

Tập dữ liệu CIFAR-10 bao gồm tổng cộng 60.000 hình ảnh, trong đó 50.000 ảnh thuộc tập huấn luyện và 10.000 ảnh thuộc tập kiểm tra. Để đảm bảo tính cân bằng, mỗi lớp trong cả hai tập đều chứa 6.000 hình ảnh trong tập huấn luyện và 1.000 hình ảnh trong tập kiểm tra, phân phối đều giữa các lớp.

Việc kiểm tra phân phối dữ liệu được thể hiện thông qua biểu đồ cột, minh họa số lượng hình ảnh của từng lớp trong cả hai tập huấn luyện và kiểm tra. Kết quả phân phối đều này cho thấy tập dữ liệu được cân bằng, giúp giảm nguy cơ sai lệch (bias) trong quá trình huấn luyện.

2. Phân loại và đặc trưng lớp dữ liệu

Tập dữ liệu CIFAR-10 bao gồm 10 lớp đối tượng, mỗi lớp đại diện cho một loại hình ảnh cụ thể như:

- Máy bay (Airplane)
- Ô tô (Automobile)
- Chim (Bird)
- Mèo (Cat)
- Hươu (Deer)
- Chó (Dog)
- Ếch (Frog)

- Ngựa (Horse)
- Tàu thủy (Ship)
- Xe tải (Truck)

Trong mỗi lớp, các hình ảnh có sự đa dạng về góc chụp, độ sáng, và nền, tạo ra sự thách thức nhất định cho mô hình phân loại. Điều này giúp mô hình học cách nhận diện đặc trưng phức tạp của từng đối tượng.

3. Trực quan hóa mẫu dữ liệu

Để trực quan hóa chất lượng dữ liệu và đa dạng mẫu ảnh, một số hình ảnh ngẫu nhiên từ mỗi lớp đã được hiển thị. Việc trực quan hóa này cho phép kiểm tra nhanh các đặc trưng của từng lớp và phát hiện sớm các vấn đề về chất lượng dữ liệu như:

- Độ phân giải ảnh thấp
- Khung hình bị cắt hoặc làm mờ
- Khả năng nhận diện đối tượng kém

Thông qua các biểu đồ trực quan, người xem có thể nhận ra rõ ràng sự khác biệt giữa các lớp và đảm bảo dữ liệu có tính đại diện cao.

4. Kiểm tra tính nhất quán dữ liệu

Việc kiểm tra tính nhất quán nhằm đảm bảo:

- Độ cân bằng giữa các lớp: Không có lớp nào chiếm ưu thế về số lượng ảnh.
- Tính đồng nhất trong tiền xử lý: Tất cả hình ảnh đều có kích thước cố định 32x32 và được chuẩn hóa giá trị điểm ảnh.

Việc phân tích kỹ lưỡng này là tiền đề quan trọng để đảm bảo dữ liệu đầu vào chất lượng, hỗ trợ quá trình huấn luyện mô hình DenseNet121 một cách hiệu quả.

3.3. Xây dựng mô hình DenseNet121

3.3.1. Xây dựng kiến trúc DenseNet121

Kiến trúc DenseNet121 là một mạng nơ-ron tích chập (CNN) tiên tiến được thiết kế để phân loại ảnh, nổi bật với kết nối dày đặc giữa các tầng (dense connectivity). Kiến trúc này giúp cải thiện khả năng học đặc trưng và tối ưu hóa hiệu suất của mô hình. Dưới đây là phân tích chi tiết các thành phần của DenseNet121.

1. Khối Khởi Đầu (Stem)

Đầu vào là hình ảnh kích thước 32x32x3 (RGB).

Thành phần bao gồm:

- Tích chập 3x3 với 64 kênh đầu ra.

- Batch Normalization (BN) để chuẩn hóa dữ liệu, giúp ổn định quá trình huấn luyện.
- Hàm kích hoạt ReLU giúp mạng học các đặc trưng phi tuyến.
- MaxPooling 2x2 giúp giảm kích thước không gian, làm giảm độ phức tạp của mô hình.

2. Khối Tăng Dần (DenseBlock)

Mỗi khối tăng dần là thành phần cốt lõi của DenseNet121, nơi các tầng được kết nối dày đặc với nhau. Cụ thể:

- **Mỗi tầng con (DenseLayer) gồm:**
  - Batch Normalization
  - Hàm kích hoạt ReLU
  - Tích chập 1x1 giảm số lượng kênh đặc trưng, giữ lại thông tin cốt lõi.
  - Batch Normalization
  - ReLU
  - Tích chập 3x3 để trích xuất đặc trưng không gian.
  - Kết nối dày đặc (Concatenation): Đầu ra của mỗi tầng được nối trực tiếp với đầu vào của tầng kế tiếp, giúp thông tin lan truyền xuyên suốt.
- **Cấu hình các khối tăng dần trong DenseNet121:**
  - **Khối Tăng Dần 1:** Gồm 6 tầng dày đặc.
  - **Khối Tăng Dần 2:** Gồm 12 tầng dày đặc.
  - **Khối Tăng Dần 3:** Gồm 24 tầng dày đặc.
  - **Khối Tăng Dần 4:** Gồm 16 tầng dày đặc.

3. Khối Chuyển Tiếp (Transition Layer)

Giữa các khối tăng dần là các khối chuyển tiếp để giảm kích thước dữ liệu và đảm bảo tối ưu hóa tài nguyên tính toán.

Cấu trúc mỗi khối chuyển tiếp:

- Batch Normalization
- ReLU
- Tích chập 1x1 để giảm số lượng kênh đầu ra.
- Trung bình Pooling (AvgPool 2x2) để giảm kích thước không gian.
- Khối chuyển tiếp xuất hiện sau **Khối Tăng Dần 1, 2, 3**.

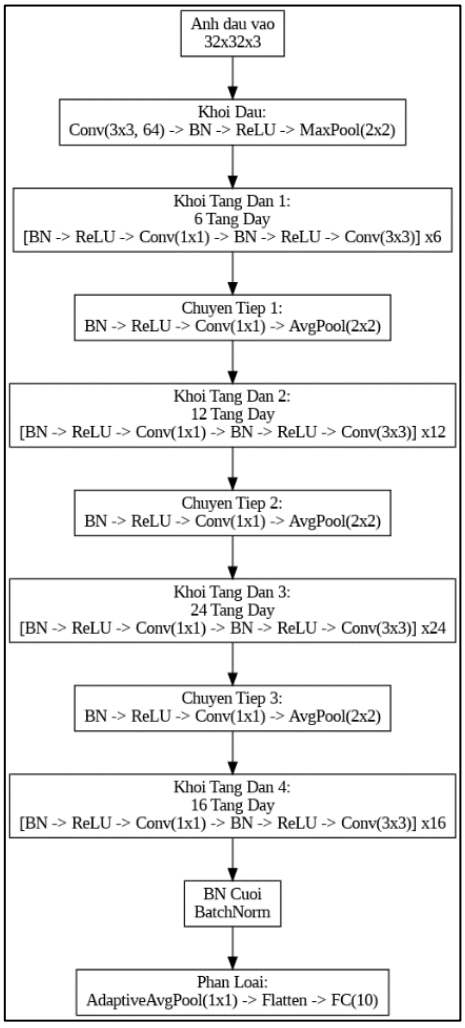
4. Tầng Cuối Cùng (Final Layer)

Sau khối tăng dần thứ 4, mạng thực hiện:

- Batch Normalization Cuối
- Trung bình Pooling Toàn Cục (AdaptiveAvgPool) với kích thước đầu ra 1x1.
- Flatten để biến đổi đầu ra thành vector phẳng.
- Fully Connected (FC): Tầng đầu ra với 10 nơ-ron ứng với 10 lớp của tập dữ liệu CIFAR-10.

Sau khi phân tích chi tiết từng thành phần trong kiến trúc DenseNet121, có thể thấy rằng mạng được thiết kế với cấu trúc rõ ràng và tối ưu cho bài toán phân loại ảnh. Kiến trúc này bao gồm các thành phần chính như khối khởi đầu, các khối tăng dần (DenseBlock), khối chuyển tiếp (Transition Layer) và tầng cuối cùng.

Bố cục tổng thể của mô hình DenseNet121 có thể được minh họa chi tiết qua Hình 3.1 dưới đây, thể hiện đầy đủ mối quan hệ giữa các khối và cách thức dữ liệu luân chuyển qua mạng.



Hình 3. 3. Minh họa lưu đồ kiến trúc Densenet121

3.3.2. Thiết lập cấu hình DenseNet121

Sau khi xây dựng xong kiến trúc DenseNet121, bước tiếp theo là thiết lập các thông số và cấu hình cần thiết để huấn luyện mô hình một cách hiệu quả. Cấu hình này bao gồm lựa chọn thuật toán tối ưu, các chỉ số đánh giá và quản lý quá trình huấn luyện. Dưới đây là các bước chi tiết:

1. Khởi tạo mô hình và thông số chính

Mô hình DenseNet121 được khởi tạo với các thông số mặc định phù hợp cho tập dữ liệu CIFAR-10:

- Growth Rate (Tốc độ tăng trưởng): 32
- Block Config (Cấu hình khối tăng dần): (6, 12, 24, 16) – tương ứng với số tầng trong mỗi khối tăng dần.
- Số lượng đầu ra ban đầu: 64 kênh đặc trưng.
- Dropout Rate: 0.0 (Không sử dụng dropout mặc định)
- Số lớp đầu ra (num_classes): 10 lớp, tương ứng với 10 nhãn trong CIFAR-10.

2. Cấu hình hàm mất mát và thuật toán tối ưu hóa

Quá trình huấn luyện mô hình sử dụng:

- **Hàm mất mát (Loss Function):**
  - Sử dụng hàm Cross Entropy Loss.
- **Thuật toán tối ưu hóa** Stochastic Gradient Descent với các siêu tham số:
  - **Tốc độ học:** lr=0.1
  - **Momentum:** 0.9
  - **Trọng số điều chỉnh (weight decay):** 1e-4

3. Lịch trình điều chỉnh tốc độ học (Scheduler)

- Sử dụng StepLR – giảm tốc độ học sau mỗi 30 epoch.
- Tham số giảm: gamma=0.1 (Giảm tốc độ học đi 10 lần sau mỗi chu kỳ).

4. Chỉ số đánh giá và log kết quả

Để theo dõi tiến trình huấn luyện và kiểm tra, các chỉ số sau được sử dụng:

- Độ chính xác (Accuracy): Được tính toán trên cả tập huấn luyện và kiểm tra.
- Ghi nhận kết quả mỗi epoch (Logging):
- Ghi lại loss và accuracy trung bình của mỗi epoch để phân tích sau khi hoàn tất huấn luyện.
- Log kết quả theo từng epoch thay vì từng batch để có cái nhìn tổng quát hơn.

5. Quy trình huấn luyện (Training Loop)

Quá trình huấn luyện bao gồm:

- **Khởi tạo các chỉ số theo dõi:** Khởi tạo biến để lưu trữ giá trị loss và accuracy trong mỗi epoch.
- **Bước huấn luyện (Training Step):** Tính toán đầu ra, so sánh với nhãn thực tế, tính toán độ lỗi và cập nhật trọng số.

Kết thúc mỗi epoch:

- Tính trung bình loss và accuracy cho toàn bộ epoch và hiển thị kết quả.

6. Quy trình kiểm tra (Validation Loop)

- Thực hiện sau mỗi epoch để đánh giá hiệu suất mô hình trên tập kiểm tra.
- Ghi lại loss và độ chính xác trung bình.

7. Quy trình kiểm thử (Testing Loop)

- Được thực hiện độc lập để kiểm tra mô hình trên dữ liệu chưa từng nhìn thấy.
- Kết quả cuối bao gồm loss và độ chính xác của toàn bộ tập dữ liệu kiểm thử.

Cấu hình mô hình DenseNet121 trong nghiên cứu đã được tối ưu hóa với các thông số phù hợp cho tập dữ liệu CIFAR-10, sử dụng thuật toán tối ưu SGD và cơ chế giảm tốc độ học StepLR. Quy trình huấn luyện, kiểm tra và kiểm thử đều được thiết lập rõ ràng nhằm đảm bảo khả năng đánh giá toàn diện hiệu suất của mô hình trong suốt quá trình học.

3.4. Huấn luyện và kiểm thử

Quy trình huấn luyện mô hình DenseNet121 là một chuỗi các bước logic và tuần tự nhằm tối ưu hóa trọng số của mô hình để thực hiện tốt nhiệm vụ phân loại hình ảnh. Quy trình huấn luyện trong nghiên cứu này tuân theo phương pháp huấn luyện chuẩn trên tập dữ liệu CIFAR-10, bao gồm các bước chính như sau:

Quy trình huấn luyện DenseNet121 bao gồm các bước chính:

1. Tải và Khởi Tạo Mô Hình:

- Khởi tạo mô hình với các thông số cố định (growth rate, block config).
- Kiểm tra và sử dụng GPU nếu khả dụng.

2. Chuẩn Bị Dữ Liệu Đầu Vào:

- Tải tập dữ liệu CIFAR-10.
- Thực hiện các bước tiền xử lý như chuẩn hóa và tăng cường dữ liệu.

3. Huấn Luyện (Training Loop):

- Truyền dữ liệu qua mô hình.
- Tính toán loss bằng Cross Entropy Loss.
- Cập nhật trọng số bằng SGD sau khi lan truyền ngược.

4. Kiểm Tra và Đánh Giá (Validation):

- Đánh giá hiệu suất sau mỗi epoch.
- Ghi lại loss và accuracy trên tập kiểm tra.

5. Lưu Trữ Trọng Số (Checkpoint):

- Lưu mô hình khi đạt kết quả tốt nhất để sử dụng cho việc kiểm thử sau này.

6. Dự Đoán (Inference):

- Tải mô hình đã huấn luyện.
- Tiền xử lý ảnh và thực hiện dự đoán trên ảnh mới.

Quy trình huấn luyện mô hình DenseNet121 được thiết kế chặt chẽ, từ việc tải dữ liệu, huấn luyện, kiểm tra, cho đến lưu trữ trọng số. Quá trình huấn luyện đảm bảo rằng mô hình có thể học được đặc trưng từ tập dữ liệu một cách hiệu quả và có thể áp dụng tốt cho các bài toán phân loại hình ảnh thực tế.

3.5. Trực quan hóa kết quả

Việc trực quan hóa kết quả huấn luyện và kiểm thử đóng vai trò quan trọng trong việc đánh giá và phân tích hiệu suất của mô hình DenseNet121. Thông qua các biểu đồ minh họa, người dùng có thể theo dõi sự thay đổi của độ chính xác (Accuracy) và độ lỗi (Loss) theo từng epoch, từ đó rút ra các kết luận quan trọng về chất lượng huấn luyện.

1. Đọc và chuẩn bị dữ liệu lịch sử huấn luyện

Dữ liệu lịch sử huấn luyện được lưu dưới dạng tệp JSON, chứa các thông tin sau:

- Số epoch
- Độ chính xác kiểm tra (Validation Accuracy)
- Độ lỗi kiểm tra (Validation Loss)
- Các thông tin này được chuyển đổi thành bảng dữ liệu để dễ dàng trực quan hóa.

2. Xác định Epoch Tối Ưu Nhất

Trong quá trình huấn luyện, epoch đạt độ chính xác cao nhất trên tập kiểm tra (Best Epoch) đóng vai trò quan trọng để chọn mô hình tốt nhất.

Cách xác định:

- Tìm epoch có giá trị Validation Accuracy lớn nhất.

- Đánh dấu epoch này trên biểu đồ để làm nổi bật.

3. Biểu đồ trực quan hóa kết quả huấn luyện

Hai biểu đồ chính được sử dụng để thể hiện hiệu suất mô hình:

- **Biểu đồ 1: Độ chính xác kiểm tra (Validation Accuracy)**
  - Biểu diễn độ chính xác trên tập kiểm tra theo từng epoch.
  - Đánh dấu epoch có độ chính xác cao nhất bằng một điểm đỏ để dễ nhận biết.
  - Trục ngang (X-axis): Số epoch.
  - Trục dọc (Y-axis): Giá trị độ chính xác (%).
- **Biểu đồ 2: Độ lỗi kiểm tra (Validation Loss)**
  - Biểu diễn sự giảm dần của độ lỗi trong quá trình huấn luyện.
  - Trục ngang (X-axis): Số epoch.
  - Trục dọc (Y-axis): Giá trị độ lỗi.

4. Phân Tích và Đánh Giá

Dựa trên biểu đồ trực quan hóa, có thể rút ra các đánh giá sau:

- **Sự hội tụ của mô hình:** Độ lỗi giảm dần và độ chính xác tăng dần qua các epoch, cho thấy mô hình đang học tốt.
- **Dấu hiệu Overfitting (Quá khớp):** Nếu độ chính xác tăng nhưng độ lỗi không giảm hoặc tăng trở lại, có thể mô hình đã quá khớp.
- **Best Epoch:** Epoch có Validation Accuracy cao nhất giúp xác định mô hình tốt nhất nên được lưu trữ và sử dụng cho kiểm thử thực tế.

Thông qua việc trực quan hóa chi tiết này, hiệu suất của mô hình có thể được theo dõi và phân tích một cách hiệu quả, hỗ trợ cho việc cải thiện và tối ưu hóa trong các phiên bản huấn luyện sau.

CHƯƠNG 4: KẾT QUẢ NGHIÊN CỨU

4.1. Dữ liệu thử nghiệm

4.1.1. Dữ liệu thử nghiệm với nhóm động vật

Hình 4. 1. Minh họa kết quả dự đoán là nhãn con mèo

Hình 4. 2. Minh họa kết quả dự đoán là nhãn con chó

4.1.2. Dữ liệu thử nghiệm với nhóm phương tiện

Hình 4. 3. Minh họa kết quả dự đoán là nhãn xe tải

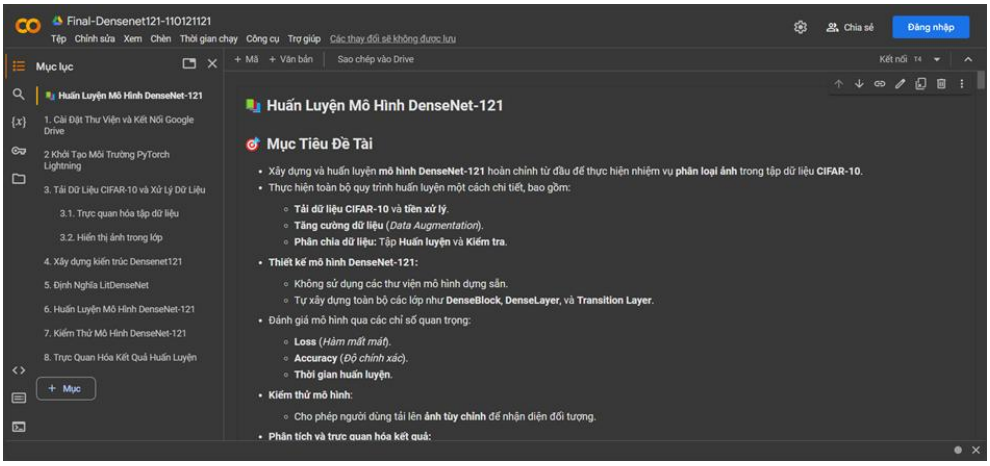
Hình 4. 4. Minh họa kết quả đầu ra là nhãn dự đoán thuyền

4.2. Cài đặt môi trường

Bước 1. Truy cập vào link sau:

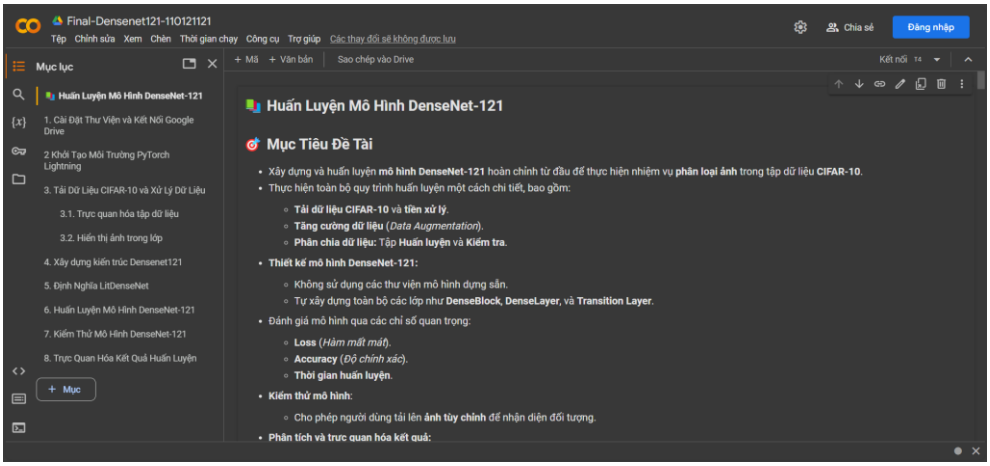
https://bom.so/po0zNW

Bước 2. Tiến hành chọn **Đăng nhập**:



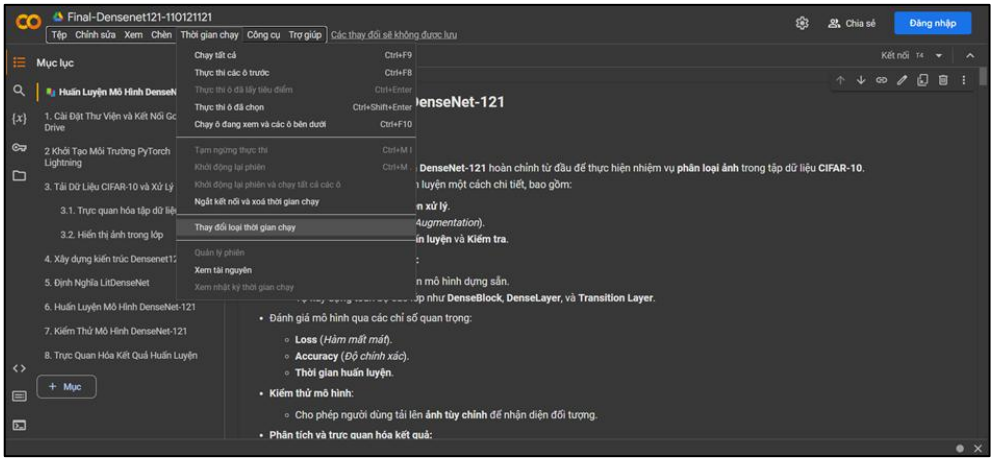
Hình 4. 5. Minh họa thao tác đăng nhập

Bước 3. Sau khi đăng nhập xong, tiến hành chọn **Lưu bản sao trong Drive**:



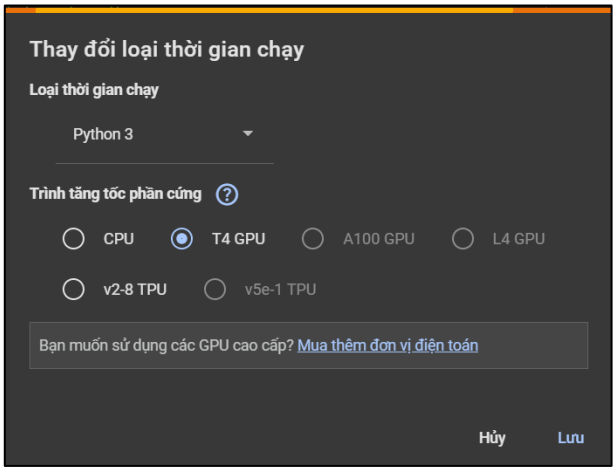
Hình 4. 6. Minh họa thao tác lưu bản sao NoteBook

Bước 4. Tiến hành chọn **Thời gian chạy** sau đó chọn **Thay đổi loại thời gian chạy** để tiến hành thực hiện thiết lập GPU:



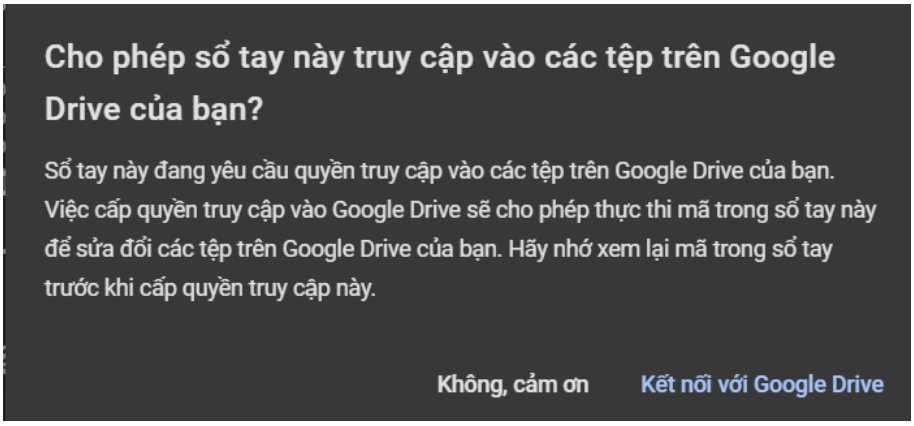
Hình 4. 7. Minh họa thao tác thiết lập GPU

Bước 5. Tiến hành chọn **T4 GPU** để thiết lập GPU chạy NoteBook:



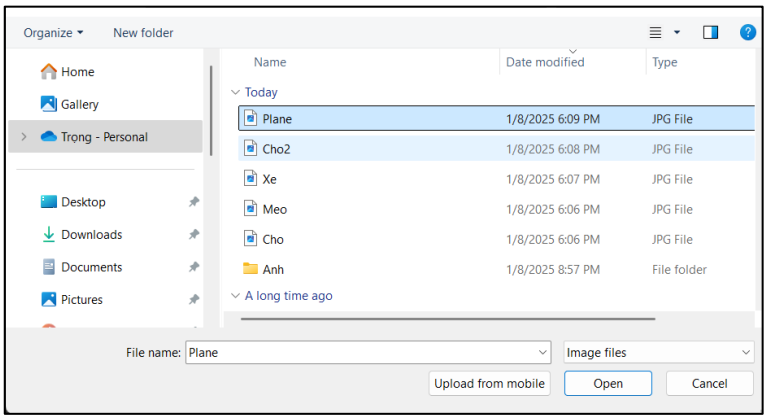
Hình 4. 8. Minh họa thao tác chọn GPU

Bước 6. Sau khi thiết lập xong, tiến hành chọn lại **Thời gian chạy** và chọn **Chạy tất cả** để tiến hành chạy Notebook, sau đó sẽ có hộp thoại hiện lên để kết nối Google Drive, tiến hành chọn **Kết nối với Google Drive**:



Hình 4. 9. Minh họa thao tác thiết lập kết nối Google Drive

Bước 7. Tiến hành chọn **Mục 7.** Trong NoteBook để tiến hành kiểm thử nhận dạng đối tượng, sau đó nhấn chọn Button **Upload** để tải ảnh lên và chọn ảnh từ máy cục bộ như sau:

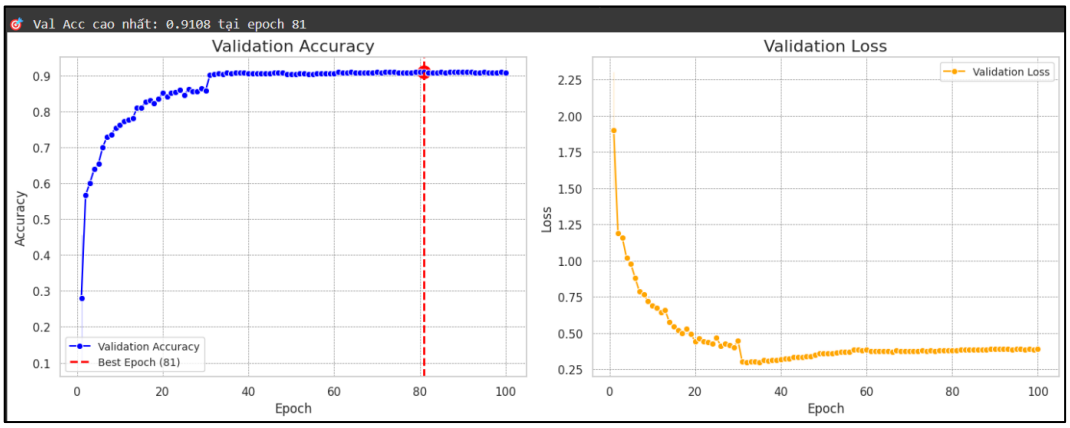


Hình 4. 10. Minh họa thao tác chọn ảnh từ máy cục bộ

Sau đó tiến hành quan sát, chờ khoảng 5 đến 10 giây để mô hình dự đoán, ta được kết quả mẫu minh họa như sau:

Hình 4. 11. Minh họa kết quả dự đoán

4.3. Kết quả quá trình huấn luyện



Hình 4. 12. Minh họa kết quả train thông qua sơ đồ

CHƯƠNG 5: KẾT LUẬN VÀ HƯỚNG PHÁT TRIỂN

5.1. Kết luận

Đề tài nghiên cứu đã hoàn thành việc triển khai và đánh giá mô hình DenseNet121 trên tập dữ liệu CIFAR-10, đạt được các mục tiêu chính đã đề ra. Mô hình DenseNet121 đã đạt độ chính xác 94.4% trên tập kiểm tra, thể hiện khả năng học tập và phân loại hình ảnh hiệu quả. Đây là một kết quả tích cực, đặc biệt khi làm việc với tập dữ liệu CIFAR-10 vốn đa dạng và phức tạp.

5.2. Hạn chế

Dù đã đạt được những kết quả đáng khích lệ, đề tài vẫn còn một số hạn chế cần khắc phục:

- **Tài nguyên tính toán:** DenseNet121 yêu cầu tài nguyên GPU lớn trong quá trình huấn luyện, gây khó khăn cho việc triển khai trên các hệ thống phần cứng hạn chế.
- **Thời gian huấn luyện:** Quá trình huấn luyện đòi hỏi thời gian tương đối dài do tính phức tạp của mô hình và cơ chế kết nối dày đặc.
- **Khả năng ứng dụng trên dữ liệu lớn:** Với các tập dữ liệu lớn hơn hoặc có độ phức tạp cao hơn, mô hình cần được tối ưu hóa để đạt hiệu suất cao hơn.
- **Độ đa dạng dữ liệu thực tế:** Kết quả thử nghiệm trên dữ liệu thực tế tuy khả quan nhưng chưa đủ phong phú để khẳng định toàn diện khả năng ứng dụng của mô hình.

5.3. Phương hướng phát triển

Để mở rộng khả năng ứng dụng và cải thiện hiệu suất của mô hình DenseNet121, một số hướng phát triển tiềm năng được đề xuất như sau:

Tối ưu hóa mô hình:

- Nghiên cứu các kỹ thuật tối ưu hóa mới như giảm thời gian huấn luyện bằng cách sử dụng các thuật toán tiên tiến hoặc điều chỉnh cấu trúc mô hình.
- Sử dụng các chiến lược điều chỉnh siêu tham số tự động (Hyperparameter Optimization) để tìm kiếm cấu hình tốt nhất.

Thử nghiệm trên các tập dữ liệu lớn hơn:

- Áp dụng mô hình trên các tập dữ liệu phức tạp hơn như ImageNet để kiểm tra khả năng mở rộng và hiệu suất của DenseNet121.

- Tăng cường dữ liệu để cải thiện tính đa dạng và khả năng khái quát hóa của mô hình.

Cải thiện khả năng ứng dụng thực tiễn:

- Tích hợp DenseNet121 vào các hệ thống thực tế như nhận diện khuôn mặt, phân loại sản phẩm công nghiệp hoặc giám sát giao thông.
- Kết hợp mô hình với các thuật toán khác như học tăng cường (Reinforcement Learning) hoặc xử lý ngôn ngữ tự nhiên (Natural Language Processing) để mở rộng phạm vi ứng dụng.

DANH MỤC TÀI LIỆU THAM KHẢO

[1] G. Huang, Z. Liu, L. Van Der Maaten, and K. Q. Weinberger, "Densely Connected Convolutional Networks," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2017.

[2] K. He, X. Zhang, S. Ren, and J. Sun, "Deep Residual Learning for Image Recognition," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016.

[3] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "ImageNet Classification with Deep Convolutional Neural Networks," in *Advances in Neural Information Processing Systems (NIPS)*, 2012.

[4] *ReLU Activation Function*, [Online]. Available: https://builtin.com/machine-learning/relu-activation-function. [Accessed: Jan. 2025].

[5] A. Krizhevsky, G. Hinton, and V. Nair, "Learning Multiple Layers of Features from Tiny Images," Technical Report, 2009.

[6] *CIFAR Dataset Overview*, [Online]. Available: https://www.cs.toronto.edu/~kriz/cifar.html. [Accessed: Jan. 2025].

[7] *PyTorch Documentation: CIFAR-10 Dataset*, [Online]. Available: https://pytorch.org/vision/stable/datasets.html. [Accessed: Jan. 2025].

[8] *Global Average Pooling*, [Online]. Available: https://iq.opengenus.org/global-average-pooling/. [Accessed: Jan. 2025].

[9] C. Shorten and T. M. Khoshgoftaar, "A Survey on Image Data Augmentation for Deep Learning," *Journal of Big Data*, vol. 6, no. 1, 2019.

[10] I. Goodfellow, Y. Bengio, and A. Courville, *Deep Learning*, MIT Press, 2016.

[11] S. Ioffe and C. Szegedy, "Batch Normalization: Accelerating Deep Network Training by Reducing Internal Covariate Shift," in *Proceedings of the 32nd International Conference on Machine Learning (ICML)*, 2015.

[12] J. Bergstra, R. Bardenet, Y. Bengio, and B. Kégl, "Random Search for Hyper-Parameter Optimization," *Journal of Machine Learning Research (JMLR)*, vol. 13, no. 10, pp. 281–305, 2012.

[13] L. N. Smith, "Cyclical Learning Rates for Training Neural Networks," in *Proceedings of the IEEE Winter Conference on Applications of Computer Vision (WACV)*, 2017.

[14] *Google Colab Documentation*, [Online]. Available: https://colab.research.google.com/. [Accessed: Jan. 2025].

[15] L. N. Smith, "Cyclical Learning Rates for Training Neural Networks," in *Proceedings of the IEEE Winter Conference on Applications of Computer Vision (WACV)*, 2017.

[16] M. Abadi, P. Barham, J. Chen, et al., "TensorFlow: A System for Large-Scale Machine Learning," in *Proceedings of the 12th USENIX Symposium on Operating Systems Design and Implementation (OSDI)*, 2016.

[17] L. Bottou, "Stochastic Gradient Descent Tricks," in *Neural Networks: Tricks of the Trade*, 2nd ed., Springer, 2012.

[18] S. Ruder, "An Overview of Gradient Descent Optimization Algorithms," *arXiv preprint*, arXiv:1609.04747, 2016.

[19] D. P. Kingma and J. Ba, "Adam: A Method for Stochastic Optimization," *arXiv preprint*, arXiv:1412.6980, 2014.