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

ĐỒ ÁN CƠ SỞ NGÀNH
HỌC KỲ I, NĂM HỌC 2024 - 2025

TÌM HIỂU MÔ HÌNH VGG11 VÀ ỨNG DỤNG

Giáo viên hướng dẫn:
ThS. Nguyễn Mộng Hiền

Sinh viên thực hiện:
Họ tên: Trương Nguyễn Hoàng Thanh
MSSV: 110121101
Lớp: DA21TTB

Trà Vinh, tháng 01 năm 2025

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

ĐỒ ÁN CƠ SỞ NGÀNH
HỌC KỲ I, NĂM HỌC 2024 - 2025

TÌM HIỂU MÔ HÌNH VGG11 VÀ ỨNG DỤNG

Giáo viên hướng dẫn:
ThS. Nguyễn Mộng Hiền

Sinh viên thực hiện:
Họ tên: Trương Nguyễn Hoàng Thanh
MSSV: 110121101
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 tiên, tôi xin gửi lời cảm ơn chân thành và sâu sắc nhất đến thầy Nguyễn Mộng Hiền, giảng viên hướng dẫn, người đã tận tình chỉ dạy và hỗ trợ tôi trong suốt quá trình thực hiện đồ án này. Thầy không chỉ truyền đạt những kiến thức chuyên môn quý giá mà còn chia sẻ những kinh nghiệm thực tiễn, giúp tôi hiểu rõ hơn về mô hình VGG11 cũng như các kiến thức nền tảng về mạng nơ-ron tích chập. Sự tận tâm và chỉ dẫn của thầy đã giúp tôi vượt qua những khó khăn và thử thách trong quá trình nghiên cứu, cũng như tự tin hơn trong từng bước thực hiện đồ án.

Tôi cũng xin gửi lời cảm ơn đến Khoa Kỹ thuật và Công nghệ, Trường Đại học Trà Vinh, đã tạo điều kiện thuận lợi cho tôi trong quá trình học tập và nghiên cứu. Những tài liệu, kiến thức cơ bản mà Khoa cung cấp đã trở thành nền tảng quan trọng, hỗ trợ tôi tiếp cận hiệu quả hơn với đồ án này.

Với những nỗ lực và sự hướng dẫn tận tình của thầy, tôi đã hoàn thành bài báo cáo đồ án cơ sở ngành. Tuy nhiên, do thời gian, kỹ năng còn hạn chế và tính phức tạp của đề tài, tôi nhận thấy đồ án vẫn còn một số điểm chưa hoàn thiện. Tôi rất mong nhận được những ý kiến đóng góp từ quý thầy cô để tiếp tục phát triển kỹ năng, nâng cao kiến thức và hoàn thiện bản thân hơn trong các nghiên cứu và đồ án sau này.

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 (Trí tuệ nhân tạo)
2	API	Application Programming Interface
3	CNN	Convolutional Neural Network
4	GPU	Graphics Processing Unit
5	ReLU	Rectified Linear Unit
6	SGD	Stochastic Gradient Descent
7	VGG	Visual Geometry Group

TÓM TẮT ĐỒ ÁN CƠ SỞ NGÀNH

Đồ án tập trung vào việc nghiên cứu và ứng dụng mô hình VGG11, một trong những kiến trúc mạng nơ-ron tích chập (CNN) nổi bật, để giải quyết bài toán phân loại hình ảnh trên tập dữ liệu CIFAR-100. Các bước thực hiện bao gồm tiền xử lý dữ liệu, tăng cường dữ liệu, xây dựng mô hình, huấn luyện và đánh giá hiệu suất. Kết quả được phân tích thông qua các chỉ số như hàm mất mát (Loss), độ chính xác (Accuracy), và ma trận nhầm lẫn (Confusion Matrix) để đo lường mức độ chính xác và khả năng tổng quát hóa của mô hình.

Phương pháp luận trong đồ án sử dụng các công cụ và thư viện mạnh mẽ như PyTorch để xây dựng và huấn luyện mô hình, kết hợp với nền tảng Google Colab nhằm tối ưu hóa hiệu suất với GPU. Quy trình huấn luyện áp dụng kỹ thuật Mixed Precision Training để tăng tốc độ và hiệu quả tính toán. Các kết quả trực quan như biểu đồ Loss, Accuracy và ma trận nhầm lẫn được sử dụng để đánh giá và so sánh hiệu suất của mô hình sau khi train.

Đồ án đã đạt được mục tiêu phân loại hình ảnh với độ chính xác cao trên tập dữ liệu CIFAR-100, đồng thời rút ra các bài học kinh nghiệm trong việc áp dụng mạng CNN vào các bài toán thực tế. Tuy nhiên, một số hạn chế như thời gian huấn luyện dài và yêu cầu tài nguyên tính toán cao vẫn còn tồn tại. Những gợi ý cải thiện trong tương lai bao gồm việc tối ưu hóa cấu trúc mô hình và thử nghiệm trên các tập dữ liệu lớn hơn hoặc phức tạp hơn.

MỞ ĐẦU

1. Lý do chọn đề tài

Trong bối cảnh trí tuệ nhân tạo (AI) và thị giác máy tính phát triển mạnh mẽ, việc nghiên cứu và ứng dụng các mô hình học sâu vào nhận dạng và phân loại hình ảnh đã trở thành một yêu cầu thiết yếu. Những tiến bộ trong lĩnh vực này mang lại nhiều ứng dụng thực tiễn, từ nhận diện khuôn mặt, phân tích hình ảnh y tế đến giám sát an ninh và sản xuất công nghiệp. Trong đó, các kiến trúc mạng nơ-ron tích chập (CNN) như VGG11 được xem là một bước đột phá, với khả năng trích xuất đặc trưng và phân loại đối tượng một cách chính xác và hiệu quả.

Đề tài "Tìm hiểu mô hình VGG11 và ứng dụng" tập trung vào nghiên cứu chi tiết cấu trúc và nguyên lý hoạt động của mô hình VGG11, đồng thời triển khai thực nghiệm trên tập dữ liệu CIFAR-100. VGG11 là một kiến trúc mạng CNN nổi bật, kết hợp giữa sự đơn giản và hiệu suất cao, giúp giải quyết hiệu quả các bài toán phân loại ảnh. Tập dữ liệu CIFAR-100, với 100 nhãn và 60.000 hình ảnh thuộc nhiều loại đối tượng khác nhau, cung cấp môi trường lý tưởng để đánh giá hiệu năng của mô hình trong các nhiệm vụ phân loại phức tạp.

Việc triển khai mô hình trên nền tảng Google Colab giúp tận dụng tối đa tài nguyên tính toán dựa trên GPU mà không cần đầu tư phần cứng chuyên dụng. Điều này không chỉ tăng tính khả thi của đề tài mà còn tối ưu hóa hiệu suất và chi phí trong suốt quá trình nghiên cứu.

2. Mục tiêu nghiên cứu

- Nghiên cứu kiến trúc và cơ chế hoạt động của mô hình VGG11, tập trung vào khả năng phân loại hình ảnh.
- Tiến hành phân tích và tiền xử lý tập dữ liệu CIFAR-100, đảm bảo dữ liệu đầu vào phù hợp với yêu cầu của mô hình VGG11.
- Triển khai và huấn luyện mô hình VGG11 trên nền tảng Google Colab, tận dụng GPU để tối ưu hóa hiệu suất.
- Đánh giá hiệu quả của mô hình thông qua các chỉ số hiệu suất, đặc biệt là độ chính xác trên tập dữ liệu CIFAR-100.

3. Phương pháp nghiên cứu

- Phân tích cấu trúc và các lớp của VGG11.
- Chuẩn hóa và chuyển đổi dữ liệu CIFAR-100 cho VGG11.

- Triển khai VGG11 trên Google Colab sử dụng ngôn ngữ lập trình Python.
- Đánh giá độ chính xác và hiệu quả của VGG11 trên CIFAR-100.

4. Đối tượng nghiên cứu

- **Mô hình VGG11:** Tập trung nghiên cứu cấu trúc, cơ chế hoạt động và khả năng ứng dụng của VGG11 trong phân loại hình ảnh.
- **Tập dữ liệu CIFAR-100:** Một tập dữ liệu gồm 60.000 hình ảnh màu thuộc 100 nhãn khác nhau, được sử dụng để kiểm tra hiệu suất phân loại của mô hình VGG11.
- **Quy trình tiền xử lý dữ liệu:** Bao gồm các kỹ thuật chuẩn hóa và tăng cường dữ liệu, đảm bảo đầu vào đạt chất lượng cao cho quá trình huấn luyện.
- **Nền tảng Google Colab:** Sử dụng để triển khai, huấn luyện và đánh giá mô hình VGG11 với sự hỗ trợ của GPU, tối ưu hóa tài nguyên tính toán.

5. Phạm vi nghiên cứu

Nghiên cứu tập trung vào cấu trúc và khả năng phân loại của mô hình VGG11 trên tập dữ liệu CIFAR-100 (100 nhãn, 60.000 ảnh), triển khai trên Google Colab với các thư viện Python (TensorFlow hoặc PyTorch), và đánh giá hiệu quả qua chỉ số độ chính xác.

CHƯƠNG 1: TỔNG QUAN

Trong bối cảnh trí tuệ nhân tạo (AI) và học sâu (Deep Learning) ngày càng phát triển, các mô hình mạng nơ-ron tích chập (CNN) đã trở thành công cụ quan trọng trong việc phân loại và nhận diện hình ảnh. VGG11, một kiến trúc CNN đơn giản nhưng hiệu quả, đã được chứng minh là mang lại kết quả vượt trội trong nhiều ứng dụng thực tế. Đề tài này tập trung vào việc nghiên cứu chi tiết mô hình VGG11 và triển khai thử nghiệm trên tập dữ liệu CIFAR-100, với mục tiêu đánh giá hiệu quả của mô hình trong các nhiệm vụ phân loại hình ảnh.

Mô hình VGG11 được triển khai trên nền tảng Google Colab, kết hợp với thư viện PyTorch để xây dựng và huấn luyện mô hình. Quy trình thực hiện bao gồm tiền xử lý dữ liệu, áp dụng các kỹ thuật tăng cường dữ liệu (Data Augmentation) và tối ưu hóa mô hình bằng các phương pháp hiện đại. Tập dữ liệu CIFAR-100, với 60.000 hình ảnh thuộc 100 nhãn khác nhau, được sử dụng để kiểm chứng khả năng phân loại của mô hình trong bối cảnh dữ liệu đa dạng và phức tạp.

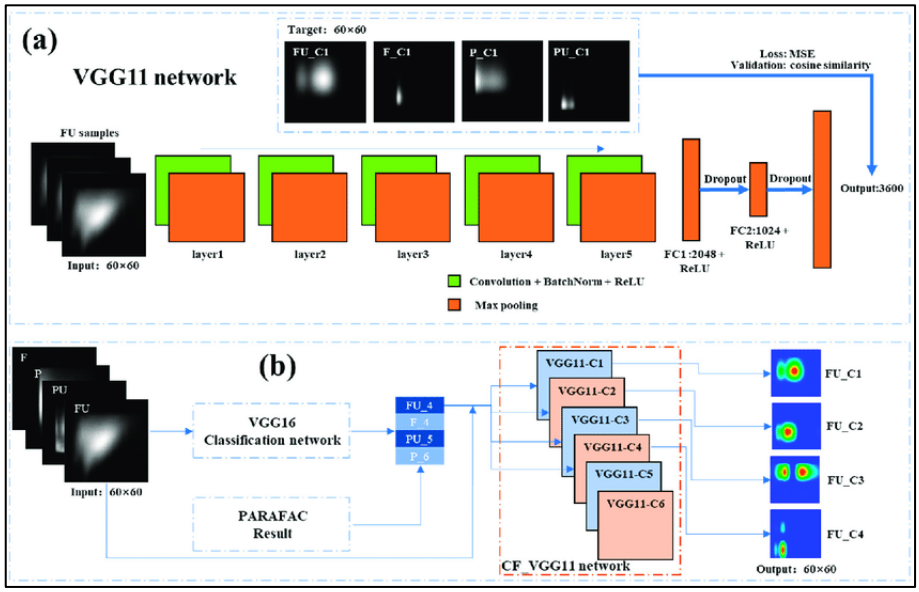
Kết quả thực nghiệm cho thấy mô hình VGG11 đạt độ chính xác 68.60% trên tập dữ liệu CIFAR-100. Thành công này khẳng định hiệu quả của mô hình trong việc phân loại hình ảnh và mở ra tiềm năng ứng dụng trong các hệ thống nhận diện đối tượng, giám sát an ninh, và phân tích hình ảnh y tế. Với thời gian huấn luyện và hiệu suất đạt được, mô hình đáp ứng tốt các yêu cầu cơ bản của các ứng dụng thực tế.

CHƯƠNG 2: NGHIÊN CỨU LÝ THUYẾT

2.1. Tổng quan mô hình VGG11

2.1.1. Lịch sử phát triển VGG11

Mô hình VGG11 là một trong những kiến trúc mạng nơ-ron tích chập (CNN) tiên tiến, được giới thiệu bởi nhóm nghiên cứu Visual Geometry Group (VGG) tại Đại học Oxford vào năm 2014. VGG11 được phát triển nhằm mục tiêu cải thiện hiệu suất của các mô hình CNN trước đó bằng cách tăng độ sâu của mạng, đồng thời duy trì cấu trúc đơn giản và dễ hiểu. Sự ra đời của VGG11 đã đánh dấu bước tiến quan trọng trong thiết kế các kiến trúc mạng nơ-ron sâu, góp phần nâng cao khả năng phân loại và nhận dạng hình ảnh trong các ứng dụng thị giác máy tính [1].



Hình 2. 1. Minh họa cấu trúc chi tiết của mô hình VGG11 [1]

Trước khi VGG11 xuất hiện, các mô hình CNN như AlexNet và ZFNet đã chứng minh tiềm năng vượt trội của học sâu trong nhiệm vụ phân loại hình ảnh trên bộ dữ liệu ImageNet. Tuy nhiên, nhóm nghiên cứu VGG nhận thấy rằng việc tăng độ sâu của mạng có thể cải thiện đáng kể hiệu suất mà không cần thay đổi cấu trúc cơ bản của các lớp tích chập và kết nối đầy đủ. Vì vậy, VGG11 được thiết kế với 11 lớp học sâu, bao gồm các lớp tích chập nhỏ (3x3) và các lớp kết nối đầy đủ, tạo nên một kiến trúc mạng có khả năng học các đặc trưng phức tạp từ dữ liệu hình ảnh [1].

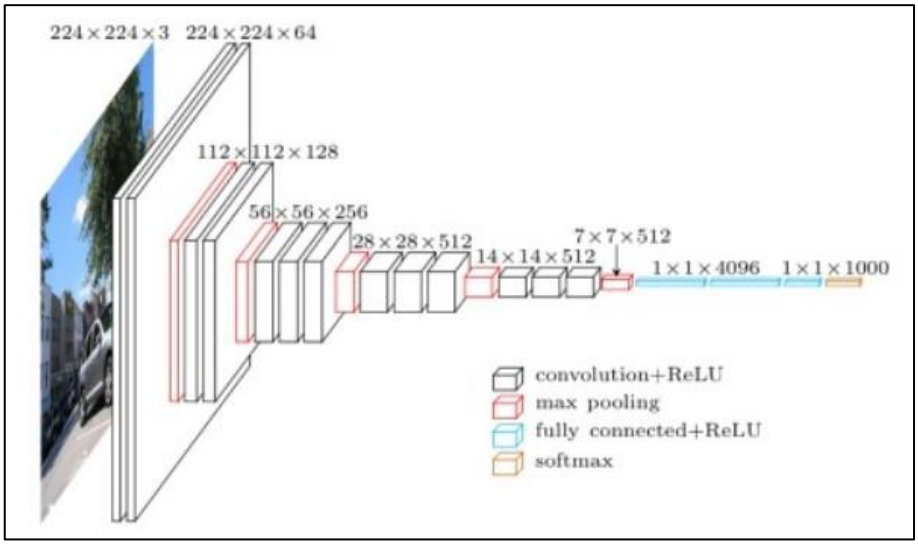
Một điểm nổi bật của VGG11 là việc sử dụng các bộ lọc nhỏ và tăng số lượng lớp tích chập để tăng cường khả năng biểu diễn của mô hình. Thiết kế này giúp VGG11 không chỉ đạt được độ chính xác cao mà còn giảm thiểu hiện tượng mất thông tin trong quá trình truyền tải dữ liệu qua các lớp mạng.

VGG11 đã trở thành cơ sở tham khảo quan trọng cho nhiều nghiên cứu trong lĩnh vực học sâu và thị giác máy tính, ảnh hưởng sâu rộng đến cách thiết kế và tối ưu hóa các mô hình mạng nơ-ron hiện đại [1].

2.1.2. Cấu trúc VGG11

Mô hình VGG11 được thiết kế với cấu trúc đơn giản nhưng hiệu quả, bao gồm 11 lớp học sâu (deep layers), trong đó có 8 lớp tích chập (convolutional layers) và 3 lớp kết nối đầy đủ (fully connected layers). Mục tiêu của thiết kế này là để tận dụng độ sâu của mạng nhằm học các đặc trưng phức tạp từ dữ liệu hình ảnh, đồng thời vẫn giữ cho kiến trúc đơn giản và dễ triển khai.

Kiến trúc của VGG11 được xây dựng dựa trên việc sử dụng các bộ lọc nhỏ có kích thước 3x3 cho tất cả các lớp tích chập. Kích thước kernel đồng nhất này không chỉ giúp đơn giản hóa quá trình triển khai mà còn làm tăng khả năng biểu diễn của mạng. Mỗi lớp tích chập sử dụng một số lượng lớn các bộ lọc (tăng dần từ 64 đến 512 bộ lọc qua các tầng mạng), với mục đích học các đặc trưng ở nhiều cấp độ khác nhau, từ các đặc trưng cục bộ (như cạnh và góc) đến các đặc trưng toàn cục phức tạp hơn. Cấu trúc của VGG11 được mô tả chi tiết như sau:



Hình 2.2. Minh họa cách hoạt động của VGG11 [2]

Lớp 1 và 2: Hai lớp tích chập đầu tiên, mỗi lớp gồm 64 bộ lọc kích thước 3x3, giúp trích xuất các đặc trưng cơ bản của hình ảnh đầu vào.

Lớp 3 và 4: Hai lớp tích chập kế tiếp với 128 bộ lọc 3x3, giúp tăng khả năng biểu diễn của mạng bằng cách học các đặc trưng ở cấp độ phức tạp hơn. Lớp max pooling tiếp tục được áp dụng sau lớp này để giảm kích thước không gian.

Lớp 5, 6, 7, và 8: Đây là bốn lớp tích chập tiếp theo, mỗi lớp bao gồm 256 bộ lọc 3x3. Các lớp này chịu trách nhiệm học các đặc trưng phức tạp hơn nữa, giúp mô hình nhận diện được các mẫu đặc trưng từ dữ liệu đầu vào. Một lớp max pooling được thêm vào sau hai lớp tích chập đầu tiên trong nhóm này để giảm thiểu kích thước.

Lớp 9 và 10: Hai lớp tích chập cuối cùng với 512 bộ lọc 3x3. Các lớp này giúp học các đặc trưng toàn cục cao cấp, cần thiết cho việc phân loại các đối tượng phức tạp trong hình ảnh. Một lớp max pooling tiếp tục được áp dụng sau hai lớp tích chập này.

Lớp kết nối đầy đủ: Sau các lớp tích chập và pooling, dữ liệu được đưa qua ba lớp kết nối đầy đủ, với số lượng nơ-ron giảm dần qua từng lớp.

Hàm kích hoạt và Softmax: Mỗi lớp tích chập và lớp kết nối đầy đủ đều sử dụng hàm kích hoạt phi tuyến ReLU, giúp tăng cường khả năng biểu diễn của mô hình. Cuối cùng, lớp kết nối đầy đủ cuối cùng sử dụng hàm softmax để chuyển đổi đầu ra thành xác suất cho mỗi lớp phân loại.

Với cấu trúc đồng nhất và độ sâu vừa phải, VGG11 đạt được hiệu suất phân loại hình ảnh tốt mà không đòi hỏi quá nhiều tham số phức tạp. Cách sử dụng các bộ lọc nhỏ 3x3 và lớp pooling giúp tối ưu hóa tài nguyên tính toán, đồng thời giảm thiểu hiện tượng overfitting trên các tập dữ liệu lớn như ImageNet. Nhờ vào cấu trúc này, VGG11 có khả năng học các đặc trưng hiệu quả, từ đặc trưng cục bộ đến các đặc trưng toàn cục cao cấp, giúp mô hình phù hợp với nhiều bài toán thị giác máy tính khác nhau.

2.1.3. So sánh VGG11 với kiến trúc khác

Mô hình VGG11, với thiết kế đơn giản và độ sâu vừa phải, đã đặt nền tảng cho nhiều kiến trúc CNN hiện đại. Tuy nhiên, so với các mô hình CNN khác như AlexNet, ResNet và Inception, VGG11 có những đặc điểm riêng biệt cả về cấu trúc lẫn hiệu suất. Phần này sẽ so sánh VGG11 với các kiến trúc CNN nổi bật khác để làm rõ điểm mạnh và hạn chế của VGG11.

So sánh với AlexNet: AlexNet là một trong những kiến trúc CNN đầu tiên nổi bật, chiến thắng trong cuộc thi ImageNet vào năm 2012 và mở ra kỷ nguyên của học sâu trong thị giác máy tính. AlexNet bao gồm 5 lớp tích chập và 3 lớp kết nối đầy đủ, với kích thước kernel lớn hơn (11x11 và 5x5) so với VGG11. Ngược lại, VGG11 sử dụng các bộ lọc nhỏ hơn (3x3) nhưng tăng số lượng lớp tích chập lên 8 lớp, điều này giúp mô hình có khả năng học các đặc trưng phức tạp hơn.

- **Ưu điểm của VGG11 so với AlexNet:** VGG11 có độ sâu lớn hơn và cấu trúc đồng nhất hơn, giúp mô hình trích xuất đặc trưng chi tiết hơn so với AlexNet.
- **Nhược điểm:** VGG11 yêu cầu tài nguyên tính toán lớn hơn so với AlexNet do số lượng lớp tích chập nhiều hơn và cần nhiều tham số hơn.

So sánh với ResNet: ResNet là một cải tiến quan trọng trong thiết kế CNN, nổi bật với khái niệm residual connections (kết nối dư), cho phép truyền thông tin qua các lớp mà không gặp vấn đề gradient vanishing (mất gradient) khi mạng càng sâu. ResNet thường được xây dựng với độ sâu cực lớn, như ResNet50, ResNet101, với các lớp kết nối dư cho phép mạng học hiệu quả hơn khi số lượng lớp tăng lên.

- **Ưu điểm của ResNet so với VGG11:** ResNet có khả năng học hiệu quả ở độ sâu lớn hơn nhờ vào các kết nối dư, giúp mô hình học các đặc trưng phức tạp mà không gặp hiện tượng mất gradient.
- **Nhược điểm của VGG11 so với ResNet:** VGG11 không có các kết nối dư, khiến mô hình gặp khó khăn hơn trong việc học các đặc trưng.

So sánh với Inception: Inception (còn gọi là GoogLeNet) là một kiến trúc CNN phức tạp hơn, sử dụng khối Inception để tối ưu hóa hiệu suất tính toán. Thay vì chỉ sử dụng các lớp tích chập đồng nhất, Inception kết hợp các bộ lọc với kích thước khác nhau trong cùng một lớp, giúp mô hình học nhiều đặc trưng khác nhau đồng thời, từ đó tăng khả năng biểu diễn của mạng mà vẫn giữ được số lượng tham số hợp lý.

- **Ưu điểm của Inception so với VGG11:** Inception có khả năng xử lý hiệu quả các đặc trưng đa dạng nhờ khối Inception, trong khi vẫn tiết kiệm tài nguyên tính toán nhờ vào thiết kế tối ưu.
- **Nhược điểm của VGG11 so với Inception:** VGG11 có cấu trúc đơn giản và đồng nhất, điều này dễ triển khai nhưng kém linh hoạt hơn trong việc xử lý nhiều loại đặc trưng khác nhau.

2.1.4. Ưu điểm và nhược điểm của VGG11

Ưu điểm nổi bật của VGG11: VGG11 được đánh giá cao nhờ vào cấu trúc đơn giản nhưng mạnh mẽ, với các bộ lọc đồng nhất 3x3 và thiết kế lớp tích chập sâu. Cấu trúc này giúp VGG11 dễ triển khai, mở rộng và hiểu rõ hơn so với các mô hình phức tạp khác, đồng thời vẫn đạt được hiệu suất cao trên các bài toán phân loại.

Nhược điểm của VGG11: Mặc dù VGG11 có khả năng biểu diễn mạnh mẽ, nhưng nó đòi hỏi tài nguyên tính toán lớn do số lượng tham số cao và thiếu các cơ chế

tối ưu hóa như kết nối dư của ResNet hoặc khối Inception. Điều này khiến VGG11 không phù hợp cho các ứng dụng có giới hạn về tài nguyên tính toán.

2.1.5. Ứng dụng thực tiễn của VGG11

Mô hình VGG11, với khả năng trích xuất đặc trưng mạnh mẽ từ dữ liệu hình ảnh, đã được áp dụng trong nhiều lĩnh vực thực tiễn, bao gồm:

Phân loại hình ảnh: VGG11 được sử dụng rộng rãi trong các bài toán phân loại hình ảnh nhờ khả năng nhận diện đặc trưng hiệu quả. Với bộ dữ liệu lớn như ImageNet, VGG11 có thể phân loại hình ảnh thành hàng ngàn danh mục khác nhau, giúp hỗ trợ các ứng dụng như tìm kiếm hình ảnh và gắn thẻ tự động trong hệ thống quản lý [1].

Nhận diện đối tượng: Kết hợp với các phương pháp khác như Region Proposal Networks (RPN), VGG11 trở thành một thành phần cốt lõi trong các hệ thống nhận diện đối tượng, như Faster R-CNN. Điều này giúp nhận diện và phân vùng các đối tượng trong ảnh hoặc video với độ chính xác cao [1].

Thị giác trong y tế: Trong y học, VGG11 đã được sử dụng để phân tích hình ảnh y tế, chẳng hạn như ảnh X-quang, MRI hoặc siêu âm. Mô hình này có thể hỗ trợ bác sĩ trong việc chẩn đoán bệnh dựa trên hình ảnh và phát hiện các dấu hiệu bất thường [2].

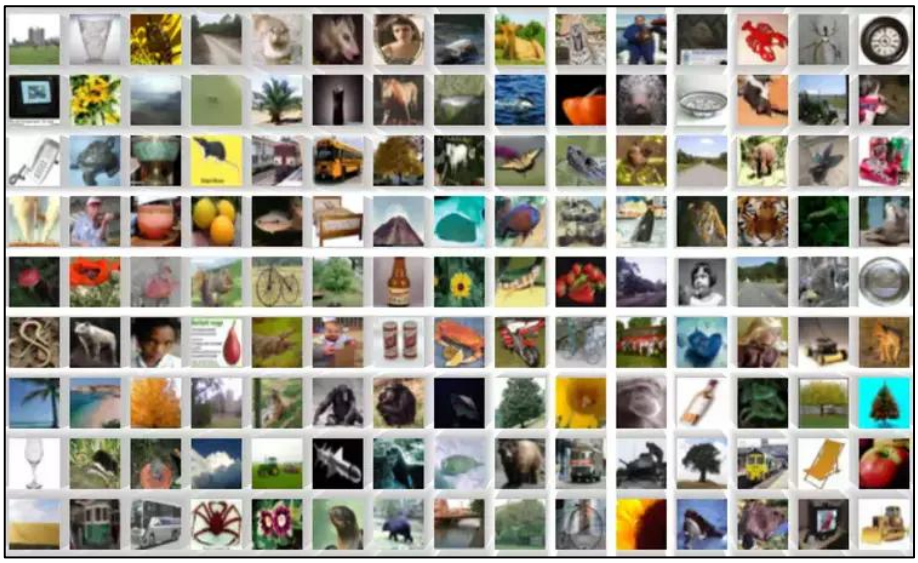
Phát hiện gian lận và an ninh: VGG11 được tích hợp trong các hệ thống an ninh để phân tích hình ảnh từ camera giám sát, nhận diện khuôn mặt hoặc phát hiện các hành vi bất thường. Trong lĩnh vực tài chính, VGG11 cũng hỗ trợ phát hiện gian lận thông qua phân tích hình ảnh chữ ký hoặc tài liệu [3].

2.2. Tổng quan bộ dữ liệu CIFAR-100

2.2.1. Giới thiệu bộ dữ liệu CIFAR-100

Bộ dữ liệu CIFAR-100 (Canadian Institute For Advanced Research) là một trong những tập dữ liệu hình ảnh quan trọng nhất được sử dụng trong lĩnh vực học sâu, đặc biệt trong các bài toán liên quan đến phân loại hình ảnh. Được phát triển bởi Alex Krizhevsky và Geoffrey Hinton, tập dữ liệu này chứa 60.000 bức ảnh màu kích thước 32x32, được tổ chức thành 100 lớp nhỏ (fine labels) thuộc 20 loại lớn (coarse labels), với mỗi lớp chứa 600 bức ảnh [5].

Điểm nổi bật của CIFAR-100 là tính đa dạng của các đối tượng trong dữ liệu. Các lớp không chỉ bao gồm các danh mục phổ biến như động vật (chim, chó, mèo) và phương tiện giao thông (xe hơi, máy bay) mà còn có các danh mục thực vật và đồ vật hàng ngày (ly, bàn, ghế).



Hình 2. 3. Minh họa bộ dữ liệu CIFAR-100 [6]

Với tính đa dạng và độ phức tạp cao, CIFAR-100 đặt ra thách thức lớn cho các mô hình học sâu trong việc xử lý hình ảnh ở độ phân giải thấp và nền phức tạp. Tập dữ liệu này được sử dụng rộng rãi để so sánh hiệu suất của các mô hình CNN nổi tiếng như AlexNet, VGG, ResNet, và EfficientNet, đồng thời hỗ trợ nghiên cứu về tăng cường dữ liệu, xử lý tiền huấn luyện và tối ưu hóa trong các ứng dụng thực tiễn [6].

2.2.2. Cấu trúc bộ dữ liệu CIFAR-100

Bộ dữ liệu CIFAR-100 được thiết kế chặt chẽ để phục vụ cho các nghiên cứu về phân loại hình ảnh, với cấu trúc cụ thể như sau:

- **Tổng số ảnh:** 60.000 bức ảnh màu RGB, kích thước 32x32 điểm ảnh.
- **Phân chia tập dữ liệu:**
    - **Tập huấn luyện:** 50.000 bức ảnh, dùng để đào tạo mô hình.
    - **Tập kiểm tra:** 10.000 bức ảnh, dùng để đánh giá hiệu suất của mô hình trên dữ liệu chưa từng thấy.
- **Hệ thống phân loại:**
    - **Loại lớn (Coarse Labels):** Gồm 20 loại như động vật, phương tiện giao thông, và thực vật.
    - **Loại nhỏ (Fine Labels):** Mỗi loại lớn được chia thành 5 loại nhỏ hơn, tổng cộng có 100 lớp.

Sự phân chia rõ ràng và cân bằng trong cấu trúc dữ liệu này làm cho CIFAR-100 trở thành một bộ dữ liệu lý tưởng để đánh giá hiệu quả của các thuật toán học sâu [5][6].

2.2.3. Đặc điểm từng lớp dữ liệu

Các lớp dữ liệu trong CIFAR-100 được xây dựng để mô phỏng sự đa dạng trong thế giới thực, với những đặc điểm chính như sau:

Tính đa dạng: Dữ liệu bao gồm nhiều danh mục khác nhau, từ động vật (như chim, cá, bò sát) đến thực vật (như cây, hoa) và các đồ vật hàng ngày (như bàn, ghế, cốc). Tính đa dạng này giúp mô hình học sâu có thể học nhiều loại đặc trưng khác nhau.

Kích thước ảnh nhỏ: Với độ phân giải chỉ 32x32 điểm ảnh, dữ liệu yêu cầu mô hình phải tối ưu khả năng trích xuất đặc trưng từ hình ảnh có chi tiết hạn chế.

Số lượng cân bằng: Mỗi lớp chứa 600 ảnh (500 ảnh trong tập huấn luyện và 100 ảnh trong tập kiểm tra). Điều này đảm bảo dữ liệu không bị lệch lớp (class imbalance), tạo điều kiện cho các mô hình học được một cách đồng đều.

Nhiễu và độ phức tạp nền: Một số bức ảnh trong CIFAR-100 chứa yếu tố gây nhiễu, như nền phức tạp hoặc chất lượng ảnh thấp. Điều này giúp kiểm tra khả năng học sâu của mô hình trong việc nhận diện đặc trưng quan trọng từ dữ liệu không hoàn hảo.

Hệ thống phân loại đa cấp: Với hai cấp nhãn (loại lớn và loại nhỏ), các bài toán có thể được triển khai từ mức độ phân loại thô (coarse-grained) đến phân loại chi tiết (fine-grained), phù hợp với nhiều mục tiêu nghiên cứu khác nhau.

Những đặc điểm này khiến CIFAR-100 trở thành một công cụ đánh giá toàn diện cho các mô hình học sâu hiện đại, đồng thời hỗ trợ kiểm tra các kỹ thuật tiên tiến như tiền xử lý dữ liệu và tăng cường dữ liệu [5][6].

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

2.3.1. Chuẩn hóa dữ liệu

Chuẩn hóa dữ liệu là một bước quan trọng trong tiền xử lý hình ảnh, đảm bảo hiệu quả và độ chính xác trong quá trình huấn luyện mô hình học sâu. Việc chuẩn hóa đưa các giá trị điểm ảnh của hình ảnh về một phạm vi nhất định, thường là [-1, 1] hoặc [0, 1], giúp cải thiện khả năng học và hội tụ của mô hình.

Vai trò của chuẩn hóa:

- **Đồng nhất hóa dữ liệu:** Chuẩn hóa giảm thiểu sự khác biệt giữa các đặc trưng, giúp các mô hình mạng nơ-ron tích chập (CNN) dễ dàng học được các đặc trưng quan trọng.
- **Tăng tốc độ hội tụ:** Dữ liệu chuẩn hóa giúp các thuật toán tối ưu hoạt động hiệu quả hơn, đặc biệt khi xử lý các tập dữ liệu lớn và phức tạp.

- **Tối ưu hóa hàm kích hoạt:** Các hàm kích hoạt như ReLU hoạt động ổn định hơn khi dữ liệu đầu vào đã được chuẩn hóa.

Phương pháp thực hiện:

Trong các tập dữ liệu hình ảnh như CIFAR-100, chuẩn hóa thường bao gồm các bước sau

- **Chuyển giá trị điểm ảnh về phạm vi [0, 1]:** Thực hiện bằng cách chia giá trị điểm ảnh cho 255.
- **Sử dụng giá trị trung bình và độ lệch chuẩn:** Tính toán giá trị mean và std cho từng kênh màu (RGB) để đưa các giá trị điểm ảnh về phân phối chuẩn.

Công thức phổ biến:

$$Pixel\ chuẩn\ hóa = \frac{Pixel\ ban\ đầu - Mean}{Std} \quad [8]$$

Đối với CIFAR-100, các giá trị mean và std thường được sử dụng là:

- **Mean:** (0.5, 0.5, 0.5)
- **Std:** (0.5, 0.5, 0.5).

Chuẩn hóa không chỉ cải thiện hiệu quả học của mô hình mà còn hỗ trợ giảm thiểu các vấn đề như mất cân bằng gradient. Trong nhiều nghiên cứu, việc chuẩn hóa đã được chứng minh là bước khởi đầu cần thiết để đạt hiệu suất cao trong các mô hình CNN, từ AlexNet, VGG, đến ResNet [7], [8].

2.3.2. Tăng cường dữ liệu

Tăng cường dữ liệu (Data Augmentation) là một kỹ thuật quan trọng trong học sâu, giúp cải thiện khả năng tổng quát hóa của mô hình bằng cách làm phong phú dữ liệu huấn luyện. Kỹ thuật này tạo ra các biến thể của dữ liệu gốc thông qua các phép biến đổi, giúp mô hình học được các đặc trưng đa dạng hơn.

Vai trò của tăng cường dữ liệu:

- **Tăng kích thước dữ liệu:** Làm giàu tập dữ liệu, đặc biệt quan trọng khi dữ liệu gốc bị hạn chế.
- **Giảm hiện tượng overfitting:** Mô hình khó học thuộc dữ liệu gốc khi được cung cấp các biến thể đa dạng.
- **Cải thiện khả năng tổng quát hóa:** Giúp mô hình hoạt động tốt hơn trên dữ liệu chưa từng thấy.

Các kỹ thuật phổ biến:

- **Lật ảnh:** Lật ảnh theo chiều ngang để tạo ra biến thể đối xứng.
- **Cắt và xoay ảnh:** Thay đổi góc nhìn của ảnh để tăng tính đa dạng.
- **Biến đổi màu sắc:** Thay đổi độ sáng, độ tương phản hoặc sắc độ.
- **Thay đổi tỷ lệ và độ phân giải:** Điều chỉnh kích thước để phù hợp với đầu vào của mô hình.

Trong bài toán phân loại hình ảnh với CIFAR-100, tăng cường dữ liệu là một phần không thể thiếu để cải thiện hiệu suất mô hình, đặc biệt là khi đối mặt với tập dữ liệu phức tạp [7], [8].

2.3.3. Phân chia dữ liệu

Phân chia dữ liệu (Train/Validation/Test Split) là một bước cơ bản nhưng vô cùng quan trọng trong quy trình học sâu. Quá trình này đảm bảo rằng mô hình được huấn luyện, đánh giá và kiểm tra một cách chính xác trên các tập dữ liệu độc lập, từ đó cải thiện khả năng tổng quát hóa của mô hình khi áp dụng vào các bài toán thực tế.

Vai trò của phân chia dữ liệu:

Tập huấn luyện (Training Set): Tập huấn luyện được sử dụng để tối ưu hóa trọng số của mô hình thông qua các thuật toán học sâu như Gradient Descent. Đây là nơi mô hình học các đặc trưng từ dữ liệu đầu vào để dự đoán chính xác nhãn đầu ra. Một tập huấn luyện đủ lớn và đa dạng sẽ giúp mô hình đạt được hiệu suất cao hơn.

Tập kiểm tra (Validation Set): Tập kiểm tra trong quá trình huấn luyện giúp đánh giá khả năng tổng quát hóa của mô hình trên dữ liệu chưa từng gặp. Tập này hỗ trợ điều chỉnh các siêu tham số như tốc độ học, độ sâu của mạng, và các thuật toán tối ưu. Sử dụng tập kiểm tra đảm bảo rằng các đặc trưng học được không bị phụ thuộc quá mức vào tập huấn luyện, giúp giảm nguy cơ overfitting.

Tập kiểm tra cuối cùng (Test Set): Tập kiểm tra cuối cùng được dùng để đo lường hiệu suất tổng thể của mô hình sau khi quá trình huấn luyện và điều chỉnh siêu tham số hoàn tất. Tập dữ liệu này là cơ sở để đánh giá mô hình trên dữ liệu thực tế, cung cấp cái nhìn khách quan về khả năng dự đoán của mô hình trong các tình huống không được chuẩn bị trước.

Quy tắc thông thường:

- Tỷ lệ phân chia dữ liệu thường được áp dụng như sau:
    - **Tập huấn luyện:** Chiếm 70-80% tổng dữ liệu.

- **Tập kiểm tra:** Chiếm 10-15%, dùng để đánh tổng quát hóa mô hình.
- **Tập kiểm tra cuối cùng:** Chiếm 10-15%, đánh giá mô hình trên dữ liệu hoàn toàn mới.

Phân chia dữ liệu đúng cách là bước đầu tiên và cũng là một trong những yếu tố quyết định đến hiệu suất và độ chính xác của mô hình học sâu. Đặc biệt, với các tập dữ liệu đa dạng và phức tạp như CIFAR-100, quá trình này không chỉ giúp cải thiện chất lượng mô hình mà còn đảm bảo kết quả đạt được có giá trị thực tiễn cao [7], [8].

2.3.4. Quản lý dữ liệu với DataLoader

Trong lĩnh vực học sâu, quản lý dữ liệu là một yếu tố quan trọng để đảm bảo quá trình huấn luyện và đánh giá mô hình diễn ra hiệu quả. Một công cụ phổ biến để xử lý dữ liệu trong PyTorch là DataLoader, giúp tự động hóa việc truy xuất, chuẩn bị, và xử lý dữ liệu theo từng batch.

Vai trò của DataLoader:

Xử lý dữ liệu theo batch: DataLoader chia dữ liệu thành các nhóm nhỏ (batch) để giảm tải bộ nhớ và tăng hiệu suất tính toán. Điều này đặc biệt hữu ích khi làm việc với các tập dữ liệu lớn như CIFAR-100.

Tăng tính ngẫu nhiên: Công cụ này hỗ trợ trộn ngẫu nhiên dữ liệu (shuffle) trong mỗi epoch, giúp mô hình không bị phụ thuộc vào thứ tự dữ liệu và cải thiện khả năng tổng quát hóa.

Hỗ trợ song song: DataLoader tận dụng nhiều luồng xử lý (multi-threading) để tải dữ liệu, giúp tăng tốc độ chuẩn bị dữ liệu trong quá trình huấn luyện.

Các chức năng chính:

- **Batch Size:** Quy định kích thước batch, ảnh hưởng đến hiệu suất tính toán và mức độ chính xác của quá trình học.
- **Shuffle:** Tính năng trộn dữ liệu ngẫu nhiên trước khi bắt đầu huấn luyện mỗi epoch, giúp mô hình không bị học theo thứ tự của dữ liệu.
- **Data Sampler:** Cho phép định nghĩa cách dữ liệu được chọn trong mỗi batch và như ưu tiên các lớp ít mẫu để giải quyết vấn đề mất cân bằng dữ liệu.

DataLoader đóng vai trò như cầu nối giữa dữ liệu thô và mô hình học sâu, đảm bảo rằng dữ liệu được cung cấp một cách nhanh chóng, hiệu quả và tối ưu hóa cho quá trình huấn luyện. Công cụ này không chỉ đơn giản hóa việc quản lý dữ liệu mà còn giúp cải thiện hiệu suất và khả năng tổng quát hóa của mô hình [7], [8].

2.4. Tổng quan về Google Colab

2.4.1. Giới thiệu Google Colab

Google Colab (Colaboratory) là một nền tảng trực tuyến miễn phí do Google phát triển, cung cấp môi trường lập trình Python trực tiếp trên trình duyệt web. Colab được thiết kế dựa trên nền tảng Jupyter Notebook, với các tính năng mở rộng như tích hợp GPU và TPU, giúp người dùng dễ dàng xây dựng, huấn luyện và triển khai các mô hình học sâu mà không cần phải đầu tư vào phần cứng mạnh mẽ [9].

Hình 2. 4. Minh họa biểu tượng Google Colab [9]

Google Colab không chỉ phổ biến trong cộng đồng học thuật và nghiên cứu, mà còn là công cụ hỗ trợ đắc lực cho những người học và phát triển trong lĩnh vực trí tuệ nhân tạo. Tích hợp với Google Drive và khả năng chia sẻ nhanh chóng thông qua liên kết, Google Colab mang lại sự linh hoạt cao trong việc cộng tác và làm việc nhóm, đặc biệt phù hợp với các dự án học sâu và thị giác máy tính [10].

2.4.2. Tính năng nỗi bật

Hỗ trợ GPU và TPU miễn phí: Một trong những tính năng mạnh mẽ nhất của Google Colab là khả năng sử dụng GPU và TPU miễn phí.

- **Với thời gian mỗi phiên làm việc như sau:**
  - **GPU:** Tối đa 4 giờ 10 phút.
  - **TPU:** Tối đa 2 giờ 10 phút.

Những giới hạn này giúp cân bằng tài nguyên giữa hàng triệu người dùng, đồng thời vẫn đảm bảo hiệu suất cao cho các bài toán học sâu [9].

Tích hợp Google Drive: Google Colab cho phép kết nối trực tiếp với Google Drive, giúp lưu trữ và truy xuất dữ liệu, mô hình huấn luyện một cách dễ dàng. Điều này giảm thiểu thời gian tải dữ liệu và tăng tính tiện lợi khi làm việc với các tệp lớn.

Hỗ trợ thư viện học sâu: Colab tích hợp sẵn nhiều thư viện phổ biến như TensorFlow, PyTorch, Keras, scikit-learn, và OpenCV, giúp người dùng tiết kiệm thời gian cài đặt và cấu hình.

Cộng tác thời gian thực: Người dùng có thể mời người khác chỉnh sửa và thực thi notebook trong thời gian thực, tương tự như Google Docs. Tính năng này rất hữu ích cho các nhóm làm việc hoặc dự án nghiên cứu chung.

Chia sẻ dễ dàng: Notebook có thể được chia sẻ qua liên kết hoặc tải xuống dưới dạng tệp .ipynb hoặc .py. Điều này hỗ trợ việc tái sử dụng và tái cấu hình mã nguồn trong các dự án khác nhau [10].

Trực quan hóa mạnh mẽ: Google Colab hỗ trợ trực tiếp các công cụ vẽ biểu đồ như Matplotlib, Seaborn, và TensorBoard, giúp người dùng dễ dàng theo dõi và phân tích kết quả mô hình.

2.4.3. Lợi ích của Google Colab

Truy cập miễn phí: Google Colab cung cấp GPU và TPU miễn phí, giúp người dùng tiếp cận công nghệ tiên tiến mà không cần đầu tư chi phí phần cứng. Đây là một lợi thế lớn, đặc biệt với những người học hoặc nghiên cứu hạn chế về ngân sách.

Dễ sử dụng: Người dùng không cần phải cài đặt môi trường lập trình phức tạp, chỉ cần đăng nhập bằng tài khoản Google và sử dụng trực tiếp trên trình duyệt.

Hỗ trợ tăng tốc tính toán: GPU và TPU giúp giảm thời gian huấn luyện mô hình đáng kể, đặc biệt với các bài toán lớn như phân loại hình ảnh hoặc xử lý ngôn ngữ tự nhiên.

Tích hợp hoàn chỉnh: Colab hoạt động liền mạch với Google Drive, giúp lưu trữ, chia sẻ, và đồng bộ hóa dữ liệu một cách nhanh chóng. Các dự án có thể được truy xuất từ bất kỳ đâu.

Thân thiện với học viên: Colab là một nền tảng lý tưởng cho việc học tập, với tài liệu phong phú và giao diện trực quan, hỗ trợ người mới bắt đầu làm quen với học sâu và các ứng dụng AI [9].

2.4.4. Hạn chế của Google Colab

Giới hạn thời gian sử dụng: Phiên bản miễn phí bị giới hạn thời gian làm việc cho mỗi phiên. GPU chỉ được sử dụng tối đa 4 giờ 10 phút mỗi lần, trong khi TPU giới hạn ở 2 giờ 10 phút. Sau thời gian này, người dùng phải khởi động lại phiên làm việc, gây gián đoạn quá trình huấn luyện.

Giới hạn tài nguyên: Dung lượng bộ nhớ và khả năng sử dụng tài nguyên phần cứng trong Colab miễn phí bị giới hạn. Điều này gây khó khăn cho các dự án yêu cầu tài nguyên lớn hoặc huấn luyện dài hạn.

Phụ thuộc vào internet: Colab yêu cầu kết nối internet ổn định để hoạt động. Điều này có thể gây bất tiện trong điều kiện không có mạng hoặc mạng yếu.

Ngắt kết nối tự động: Nếu không có hoạt động trong một khoảng thời gian nhất định, phiên làm việc sẽ tự động ngắt kết nối. Điều này làm gián đoạn các quá trình huấn luyện dài hạn.

Hạn chế trong triển khai quy mô lớn: Mặc dù hỗ trợ GPU và TPU, Colab không phù hợp để triển khai các dự án hoặc các bài toán yêu cầu tài nguyên lớn [10].

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-100 bằng mô hình VGG11. Mục tiêu chính là xây dựng một mô hình học sâu có khả năng nhận diện và phân loại chính xác các đối tượng từ ảnh, thuộc 100 lớp khác nhau như động vật, thực vật, đồ vật, và các đối tượng cụ thể khác

Hình ảnh đầu vào (Input) của bài toán là các ảnh màu RGB kích thước 32×32 điểm ảnh, được trích xuất từ tập dữ liệu CIFAR-100, minh họa tương tự Hình 3.1:

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 (Output) là một nhãn tương ứng, minh họa như Hình 3.2 dưới đây:

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

3.2. Tiền xử lý dữ liệu CIFAR-100

3.2.1. Tải và phân tích tập dữ liệu với PyTorch

CIFAR-100 là tập dữ liệu tiêu chuẩn, cung cấp 60.000 hình ảnh RGB có kích thước 32×32 điểm ảnh, được chia thành tập huấn luyện (50.000 ảnh) và tập kiểm tra (10.000 ảnh). Để chuẩn bị dữ liệu, cần xác định các đường dẫn chính:

- **`TRAIN_PATH`** chứa tập huấn luyện.
- **`TEST_PATH`** chứa tập kiểm tra.
- **`META_PATH`** chứa thông tin về nhãn các lớp.

Thiết lập hàm **`download_and_extract_cifar100`** để tự động kiểm tra, tải và giải nén dữ liệu nếu chưa tồn tại. Kết quả sau đó được xác minh bằng cách kiểm tra thư mục chứa dữ liệu, đảm bảo tất cả tệp cần thiết đều có sẵn.

Sau khi tải, quá trình phân tích tập dữ liệu bắt đầu bằng việc sử dụng hàm len để đếm số lượng hình ảnh trong từng tập, từ đó đảm bảo tính cân đối giữa tập huấn luyện và kiểm tra. Dữ liệu hình ảnh được kiểm tra cấu trúc, xác nhận mỗi ảnh với các tiêu chí như sau:

- Kích thước ảnh: (32, 32, 3).
- Giá trị điểm ảnh nằm trong phạm vi: [0, 255].
- Trích xuất các nhãn từ tệp meta và hiển thị.

Cuối cùng, dữ liệu được phân tích thêm qua việc hiển thị ngẫu nhiên các hình ảnh bằng hàm **`random.randint`** kết hợp với **`matplotlib`**, giúp trực quan hóa nội dung và chất lượng hình ảnh. Biểu đồ số lượng ảnh trong từng lớp được tạo bằng **`plt.bar`**, đảm bảo không có mất cân bằng nghiêm trọng. Ngoài ra, hàm **`np.histogram`** hoặc **`plt.hist`** được sử dụng để phân tích và trực quan hóa phân phối giá trị điểm ảnh của hình ảnh, cung cấp cơ sở cho các bước chuẩn hóa dữ liệu tiếp theo.

3.2.2. Chuẩn hóa dữ liệu với NumPy

Dữ liệu trong tập CIFAR-100 ban đầu có giá trị điểm ảnh nằm trong khoảng [0, 255], không tối ưu cho các phép toán trong mạng nơ-ron. NumPy được sử dụng để chuẩn hóa giá trị điểm ảnh về khoảng [-1, 1] thông qua hàm **`normalize_data`**, với các thông số đơn giản hóa như sau:

- **Trung bình (μ)** = **0.5** cho tất cả các kênh màu R, G, B.
- **Độ lệch chuẩn (σ)** = **0.5** cho tất cả các kênh màu R, G, B.

Sự đơn giản hóa này giúp giảm thiểu độ lệch giữa các giá trị điểm ảnh, đảm bảo dữ liệu có phân phối ổn định và phù hợp hơn cho quá trình huấn luyện mô hình.

3.2.3. Tăng cường dữ liệu với Transforms

Đối với tập huấn luyện: Tập này được tăng cường dữ liệu nhằm tạo ra sự đa dạng trong đầu vào và giảm nguy cơ overfitting. Các phép biến đổi dữ liệu được áp dụng như sau:

- **Chuyển đổi sang ảnh PIL**: Chuyển định dạng ảnh số sang PIL để hỗ trợ xử lý nâng cao.
- **Thay đổi kích thước:** Đảm bảo kích thước đồng nhất 32×32 điểm ảnh.
- **Lật ngang ngẫu nhiên:** Lật ngang ảnh với xác suất 50% để tăng tính đa dạng.
- **Cắt ngẫu nhiên:** Cắt ảnh với viền đệm 4 điểm ảnh, đảm bảo tính bất biến vị trí.

- **Chuyển đổi sang Tensor:** Chuyển ảnh thành Tensor phù hợp với PyTorch.
- **Chuẩn hóa giá trị điểm ảnh:** Chuẩn hóa giá trị điểm ảnh về [-1, 1] với trung bình và độ lệch chuẩn 0.5 cho mỗi kênh RGB.

Những bước này giúp mô hình học được nhiều đặc trưng của dữ liệu, giảm phụ thuộc vào các chi tiết cụ thể của từng ảnh, từ đó cải thiện khả năng tổng quát hóa.

Đối với tập kiểm tra: Tập này được xử lý với các bước tối giản, tập trung vào việc chuẩn hóa để giữ tính nhất quán khi đánh giá hiệu suất của mô hình. Các bước thực hiện như sau:

- **Chuyển đổi sang ảnh PIL:** Giống tập huấn luyện, ảnh được chuyển từ định dạng số sang PIL.
- **Thay đổi kích thước:** Đảm bảo kích thước chuẩn là 32×32 điểm ảnh.
- **Chuyển đổi sang Tensor:** Chuyển ảnh sang định dạng Tensor phù hợp với PyTorch.
- **Chuẩn hóa giá trị điểm ảnh:** Áp dụng phương pháp chuẩn hóa với trung bình và độ lệch chuẩn đều là 0.5.

Sự khác biệt giữa tập huấn luyện và tập kiểm tra nằm ở mức độ biến đổi dữ liệu. Trong khi tập huấn luyện được tăng cường với nhiều phép biến đổi ngẫu nhiên để tăng sự đa dạng, tập kiểm tra chỉ được chuẩn hóa và giữ nguyên cấu trúc để đảm bảo đánh giá mô hình công bằng và chính xác trên dữ liệu chưa từng thấy.

3.2.4. Phân chia tập dữ liệu

Để tối ưu hóa quá trình huấn luyện và đánh giá, tập dữ liệu CIFAR-100 được chia thành ba phần: tập huấn luyện, tập kiểm tra, và tập kiểm định (validation). Mỗi phần đóng vai trò quan trọng trong việc đảm bảo mô hình học được đặc trưng của dữ liệu và đạt hiệu quả tốt nhất. Đồng thời các tập dữ liệu được phân chia như sau:

Tập huấn luyện (Training Set): Chiếm 90% dữ liệu từ tập huấn luyện ban đầu, được sử dụng để cập nhật trọng số của mô hình thông qua quá trình tối ưu hóa.

Tập kiểm định (Validation Set): Chiếm 10% dữ liệu từ tập huấn luyện, được sử dụng để đánh giá hiệu suất mô hình trong suốt quá trình huấn luyện, giúp tránh hiện tượng overfitting.

Tập kiểm tra (Testing Set): Bao gồm toàn bộ dữ liệu từ tập kiểm tra ban đầu, đóng vai trò đánh giá cuối cùng nhằm để đo lường hiệu suất mô hình trên dữ liệu chưa từng thấy.

Quy trình thực hiện:

1. Xây dựng Dataset tùy chỉnh:

- Thiết lập lớp `CIFAR100Dataset` quản lý dữ liệu hình ảnh và nhãn tương ứng.
- Lớp này hỗ trợ áp dụng các hàm xử lý ảnh (transforms) trước khi dữ liệu được đưa vào mô hình.

2. Phân chia tập huấn luyện:

- Tập huấn luyện được chia ngẫu nhiên thành tập huấn luyện và tập kiểm định với tỷ lệ 90:10, đảm bảo tính cân đối.
- Việc chia được thực hiện thông qua hàm **`random_split`** từ PyTorch.

3. Tạo DataLoader:

- Mỗi tập dữ liệu được chuyển thành DataLoader, hỗ trợ việc xử lý dữ liệu theo từng batch và tối ưu hóa hiệu suất tính toán.
- Đồng thời, tập huấn luyện được thiết lập chế độ xáo trộn (shuffle) để tăng tính ngẫu nhiên, trong khi tập kiểm định và tập kiểm tra giữ nguyên thứ tự để đảm bảo tính nhất quán.

Việc phân chia tập dữ liệu rõ ràng và hợp lý đảm bảo rằng mô hình không chỉ học tốt từ dữ liệu mà còn đạt hiệu quả cao khi áp dụng vào các bài toán thực tế.

3.3. Xây dựng kiến trúc mô hình VGG11

3.3.1. Thiết kế kiến trúc mô hình VGG11

Kiến trúc VGG11 được chia thành hai thành phần chính: Khối trích xuất đặc trưng và khối phân loại, mỗi thành phần đảm nhận một nhiệm vụ cụ thể trong việc xử lý dữ liệu hình ảnh và dự đoán kết quả.

Dưới đây là phân tích chi tiết từng khối:

1. Khối trích xuất đặc trưng (Feature Extractor)

- **Block 1:**
    - **Tích chập:** Áp dụng 64 bộ lọc 3×3 lên dữ liệu đầu vào (32×32×3), giúp trích xuất các đặc trưng cơ bản như cạnh và góc.
    - **Chuẩn hóa Batch:** Giúp tăng tốc độ hội tụ của mô hình và ổn định quá trình huấn luyện.
    - **Hàm kích hoạt ReLU:** Kích hoạt phi tuyến để tăng khả năng biểu diễn các đặc trưng phức tạp hơn.

    - **Gộp cực đại (MaxPooling):** Giảm kích thước không gian từ 32×32 xuống 16×16, đồng thời giữ lại các đặc trưng quan trọng nhất.
- **Block 2:**
    - **Tích chập:** Áp dụng 128 bộ lọc 3×3, tăng cường khả năng nhận diện các đặc trưng chi tiết hơn như các đường cong và hình dạng.
    - **Gộp cực đại:** Kích thước không gian tiếp tục giảm từ 16×16 xuống 8×8, giúp giảm bớt độ phức tạp tính toán.
- **Block 3:**
    - **Hai lớp tích chập:** Mỗi lớp áp dụng 256 bộ lọc 3×3, cho phép mô hình học được các đặc trưng ngày càng chi tiết hơn.
    - **Chuẩn hóa Batch và ReLU:** Mỗi lớp đều được chuẩn hóa Batch và kích hoạt bằng ReLU để giữ ổn định và nâng cao hiệu quả học.
    - **Gộp cực đại:** Kích thước không gian giảm từ 8×8 xuống 4×4, giúp tinh giản lượng dữ liệu mà vẫn giữ được đặc trưng cần thiết.
- **Block 4 và Block 5:**
    - **Hai lớp tích chập:** Mỗi lớp sử dụng 512 bộ lọc 3×3, học các đặc trưng cấp cao hơn từ hình ảnh như cấu trúc tổng thể hoặc mối quan hệ giữa các phần trong hình ảnh.
    - **Chuẩn hóa Batch và ReLU:** Tiếp tục duy trì sự ổn định và hiệu quả huấn luyện thông qua chuẩn hóa và kích hoạt phi tuyến.
    - **Gộp cực đại:** Kích thước không gian giảm dần từ 4×4 xuống 1×1 ở cuối Block 5, tạo ra một bản đồ đặc trưng nhỏ gọn nhưng chứa đầy đủ thông tin cần thiết.

Tóm tắc quy trình:

- Từ Block 1 đến Block 5, số lượng kênh được tăng dần từ 64 lên 512, đồng thời kích thước không gian giảm từ 32×32 xuống 1×1.
- Mỗi Block được thiết kế để học các đặc trưng từ đơn giản đến phức tạp, từ các đường nét cơ bản đến các chi tiết cụ thể và cấu trúc tổng thể của hình ảnh.
- Việc kết hợp chuẩn hóa Batch, ReLU, và MaxPooling đảm bảo hiệu quả huấn luyện và giảm thiểu overfitting.

Khối trích xuất đặc trưng không chỉ học được các thông tin cần thiết mà còn làm giảm bớt dữ liệu dư thừa, tạo điều kiện thuận lợi cho khối phân loại phía sau hoạt động hiệu quả hơn.

2. Khối phân loại (Classifier)

Khối phân loại là giai đoạn cuối cùng trong kiến trúc VGG11, sử dụng các lớp fully connected để dự đoán kết quả dựa trên các đặc trưng đã được trích xuất từ khối đặc trưng. Khối này bao gồm các bước chính sau:

Bước đầu tiên trong khối này là, làm phẳng dữ liệu (Flattening):

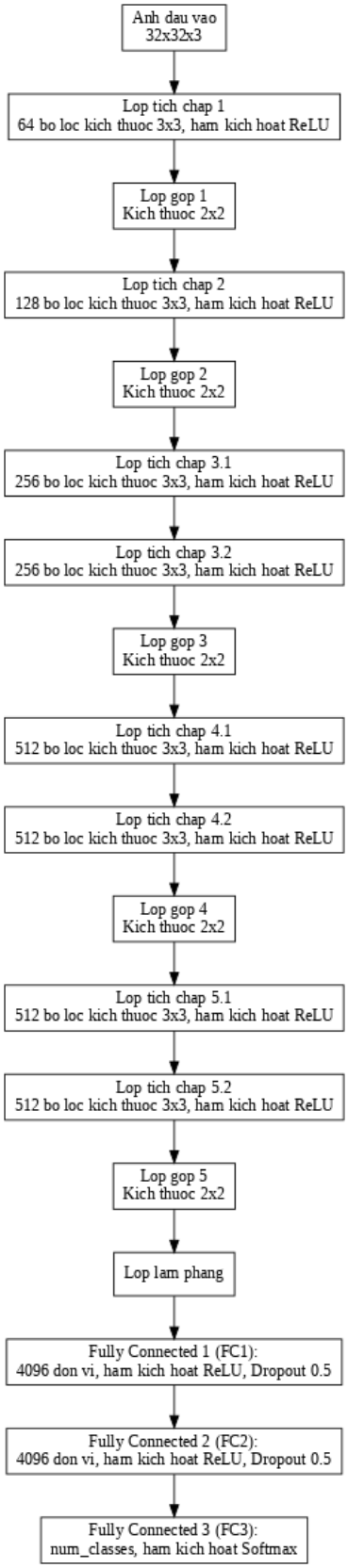
- Đầu ra từ khối đặc trưng có dạng 512×1×1 (512 kênh với kích thước không gian 1×1).
- Kết quả này được làm phẳng thành một vector đơn chiều có độ dài 512 để chuẩn bị cho việc xử lý bởi các lớp fully connected.

Tiếp theo đó là đến các lớp Fully Connected (FC):

- **Lớp Fully Connected 1 (FC1):**
  - Gồm 4096 đơn vị, có nhiệm vụ học và kết hợp các đặc trưng chi tiết từ đầu ra của khối đặc trưng.
  - Dropout với tỷ lệ 50% được áp dụng để giảm nguy cơ overfitting bằng cách ngẫu nhiên loại bỏ một số kết nối trong quá trình huấn luyện.
  - Hàm kích hoạt ReLU được sử dụng để tăng khả năng học phi tuyến.
- **Lớp Fully Connected 2 (FC2):**
  - Tương tự FC1, FC2 cũng gồm 4096 đơn vị, tiếp tục tinh chỉnh và tổng hợp các đặc trưng từ lớp trước.
  - Dropout 50% và hàm ReLU tiếp tục được sử dụng để đảm bảo tính ổn định và khả năng tổng quát hóa của mô hình.
- **Lớp Fully Connected 3 (FC3):**
  - Đây là lớp cuối cùng trong khối phân loại, có số đầu ra tương ứng với số lớp cần phân loại (100 lớp trong tập CIFAR-100).
  - Sử dụng hàm Softmax để chuyển đổi đầu ra thành xác suất, giúp mô hình đưa ra dự đoán cho từng lớp.

Khối phân loại đóng vai trò quan trọng trong việc biến các đặc trưng được trích xuất thành kết quả dự đoán cuối cùng, đảm bảo mô hình có thể tổng quát hóa tốt trên dữ liệu kiểm tra.

Dưới đây là hình minh họa lưu đồ kiến trúc VGG11, thể hiện chi tiết các khối trong mô hình từ giai đoạn trích xuất đặc trưng đến khối phân loại, giúp trực quan hóa quy trình xử lý dữ liệu và dự đoán kết quả:



Hình 3. 3. Minh họa kiến trúc VGG11.

3.3.2. Triển khai mô hình VGG11

Quá trình triển khai mô hình VGG11 được xây dựng một cách logic, tuần tự và kết hợp chặt chẽ giữa các bước. Mỗi bước đều đóng vai trò quan trọng trong việc cấu hình mô hình, huấn luyện hiệu quả và đạt được kết quả tốt nhất trên tập dữ liệu CIFAR-100. Các bước cụ thể bao gồm:

1. Khởi tạo mô hình

Mô hình VGG11 được khởi tạo với 100 lớp đầu ra, tương ứng với số lớp trong tập dữ liệu CIFAR-100. Bằng cách sử dụng lệnh .to(device), mô hình được chuyển lên GPU để tăng tốc quá trình tính toán và huấn luyện. Điều này đảm bảo mô hình có thể xử lý hiệu quả khối lượng dữ liệu lớn và các phép tính phức tạp.

Khâu khởi tạo là nền tảng cho toàn bộ quá trình triển khai, giúp mô hình sẵn sàng để thực hiện các bước huấn luyện tiếp theo.

2. Định nghĩa hàm mất mát

Hàm mất mát `CrossEntropyLoss` được lựa chọn nhằm đo lường sự khác biệt giữa nhãn dự đoán của mô hình và nhãn thực tế. Đây là hàm mất mát phổ biến và tối ưu cho các bài toán phân loại đa lớp.

Vai trò: Tạo ra tín hiệu phản hồi, giúp mô hình điều chỉnh trọng số trong các lần huấn luyện kế tiếp.

Ý nghĩa: Đảm bảo mô hình học đúng hướng, cải thiện hiệu suất phân loại.

3. Cấu hình bộ tối ưu hóa

Bộ tối ưu hóa Stochastic Gradient Descent (SGD) được thiết lập với các thông số quan trọng:

- **Learning rate (lr = 0.01):** Tốc độ học của mô hình.
- **Momentum (0.9):** Giúp giảm các dao động không cần thiết trong quá trình cập nhật trọng số.
- **Weight decay (5e-4):** Áp dụng regularization để điều chỉnh các trọng số, giúp giảm hiện tượng overfitting và cải thiện khả năng tổng quát hóa của mô hình.

Sự kết hợp giữa các thông số này giúp mô hình hội tụ nhanh, ổn định và đạt hiệu suất tối ưu.

4. Thiết lập lịch trình điều chỉnh tốc độ học

Lịch trình giảm tốc độ học CosineAnnealingLR được sử dụng để giảm dần learning rate theo hàm cosine từ giá trị ban đầu (0.01) về gần 0 trong suốt 400 epoch.

Công dụng của CosineAnnealingLR:

- Cho phép mô hình học nhanh ở giai đoạn đầu.
- Đảm bảo mô hình hội tụ tốt ở giai đoạn cuối.

5. Gradient Scaling cho Mixed Precision Training

Được thiết lập nhằm để tối ưu hóa thời gian huấn luyện và sử dụng hiệu quả tài nguyên phần cứng, Gradient Scaling được triển khai thông qua GradScaler.

Công dụng của Gradient Scaling:

- **Tăng tốc độ:** Rút ngắn thời gian xử lý mà vẫn đảm bảo độ chính xác tính toán.
- **Giảm lỗi số học:** Khắc phục các vấn đề liên quan đến giá trị rất nhỏ trong gradient.
- **Tối ưu bộ nhớ:** Giảm yêu cầu về bộ nhớ khi huấn luyện trên GPU.

Gradient Scaling là một giải pháp thiết thực để tăng hiệu suất huấn luyện mà không ảnh hưởng đến chất lượng mô hình.

3.4. Huấn luyện và đánh giá mô hình

3.4.1. Thiết lập hàm huấn luyện và đánh giá

Trong quá trình huấn luyện mô hình VGG11, việc tối ưu hóa trọng số theo từng epoch là bước quan trọng để cải thiện hiệu suất mô hình. Hàm `train_epoch` được triển khai nhằm đảm bảo các trọng số được cập nhật hiệu quả thông qua quy trình huấn luyện chi tiết. Một đặc điểm nổi bật là áp dụng kỹ thuật Mixed Precision Training, giúp tăng tốc độ xử lý và giảm tiêu tốn bộ nhớ GPU.

Quy trình huấn luyện mỗi epoch bao gồm:

- **Bật chế độ huấn luyện:** Mô hình được đặt ở chế độ `train()` để kích hoạt quá trình cập nhật trọng số.
- **Xử lý dữ liệu:** Các batch dữ liệu từ tập huấn luyện được chuyển đến thiết bị tính toán (GPU/CPU). Dữ liệu đầu vào đi qua các lớp trong mô hình để tính toán đầu ra dự đoán.
- **Gradient Scaling:** Kỹ thuật autocast được sử dụng để thực hiện tính toán với độ chính xác hỗn hợp (Mixed Precision). Sau đó, GradScaler giúp mở rộng gradient nhằm đảm bảo độ ổn định số học.
- **Tính toán lỗi:** Hàm mất mát criterion (CrossEntropyLoss) đo lường sự khác biệt giữa đầu ra dự đoán và nhãn thực tế.

- **Cập nhật trọng số:** `optimizer.step()` được gọi để điều chỉnh trọng số dựa trên gradient đã tính toán.

Kết thúc mỗi epoch, giá trị trung bình của lỗi huấn luyện được ghi lại, phục vụ việc phân tích kết quả. Quy trình này không chỉ đảm bảo tính hiệu quả mà còn giảm thiểu hiện tượng overfitting nhờ các biện pháp như dropout và chuẩn hóa batch.

3.4.2. Lưu trạng thái mô hình

Để đảm bảo mô hình đạt hiệu quả trên dữ liệu chưa từng thấy, việc đánh giá hiệu suất trên tập validation là rất cần thiết. Hàm `evaluate` được triển khai với mục tiêu đo lường hai chỉ số quan trọng: `loss` trung bình và độ chính xác.

Cách hoạt động của hàm `evaluate`:

- **Chuyển sang chế độ đánh giá:** Mô hình được đặt ở chế độ `eval()` để vô hiệu hóa các kỹ thuật như `dropout` nhằm giữ kết quả ổn định.
- **Không tính gradient:** `torch.no_grad()` được sử dụng để tiết kiệm bộ nhớ và tăng tốc độ xử lý khi đánh giá.
- **Tính toán hiệu suất:**
  - **Loss trung bình:** Được tính bằng cách áp dụng hàm mất mát criterion trên từng batch dữ liệu.
  - **Độ chính xác:** Dựa trên số lượng dự đoán đúng so với tổng số mẫu trong tập validation.
  - **Kết quả:** Giá trị loss trung bình và độ chính xác (%) được trả về sau khi hoàn tất đánh giá trên toàn bộ tập validation.

Quy trình này giúp xác định xem mô hình có khả năng tổng quát hóa tốt hay không và cung cấp thông tin quan trọng để điều chỉnh các siêu tham số.

3.4.3. Phân tích kết quả huấn luyện và đánh giá

Hàm huấn luyện chính là sự kết hợp của **`train_epoch`** và **`evaluate`**, được thực hiện trong vòng lặp qua nhiều epoch. Quy trình này không chỉ tối ưu hóa mô hình mà còn ghi nhận hiệu suất và lưu trạng thái tốt nhất.

Các bước thực hiện trong hàm huấn luyện:

- **Bắt đầu epoch:** Ghi nhận thời gian bắt đầu huấn luyện.
- **Huấn luyện trên tập train:** `Hàm train_epoch` được gọi để huấn luyện mô hình qua từng batch. Kết thúc, loss trung bình của epoch được ghi nhận.

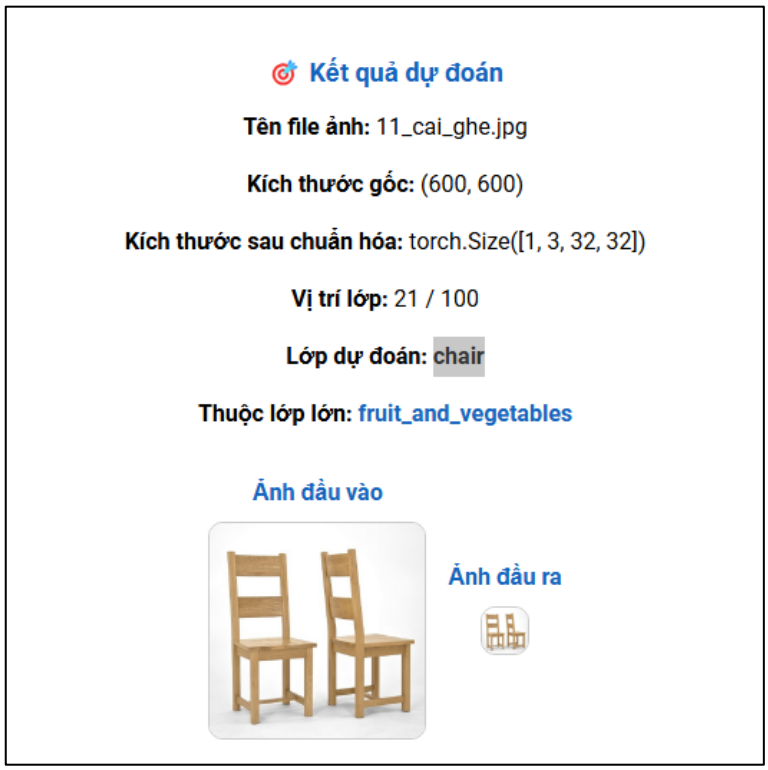
- **Đánh giá trên tập validation:** Hàm evaluate được sử dụng để tính toán loss và độ chính xác trên tập validation.
- **Lưu trạng thái mô hình:** Nếu mô hình đạt độ chính xác cao nhất trên tập validation, trạng thái mô hình (checkpoint) sẽ được lưu. Trạng thái bao gồm:
  - Trọng số mô hình.
  - Trạng thái optimizer và scheduler.
  - Epoch hiện tại.
- **Điều chỉnh learning rate:** Sử dụng `scheduler.step()` để giảm tốc độ học theo từng epoch, đảm bảo quá trình hội tụ ổn định.

Việc lưu trạng thái mô hình không chỉ giúp đảm bảo không mất dữ liệu huấn luyện mà còn cho phép tiếp tục huấn luyện hoặc sử dụng mô hình tại trạng thái tốt nhất mà không cần chạy lại từ đầu.

CHƯƠNG 4: KẾT QUẢ NGHIÊN CỨU

4.1. Dữ liệu thử nghiệm tổng quan

4.1.1. Dữ liệu thử nghiệm nhóm đồ vật



Hình 4. 1. Minh họa kết quả dự đoán là nhãn cái ghế

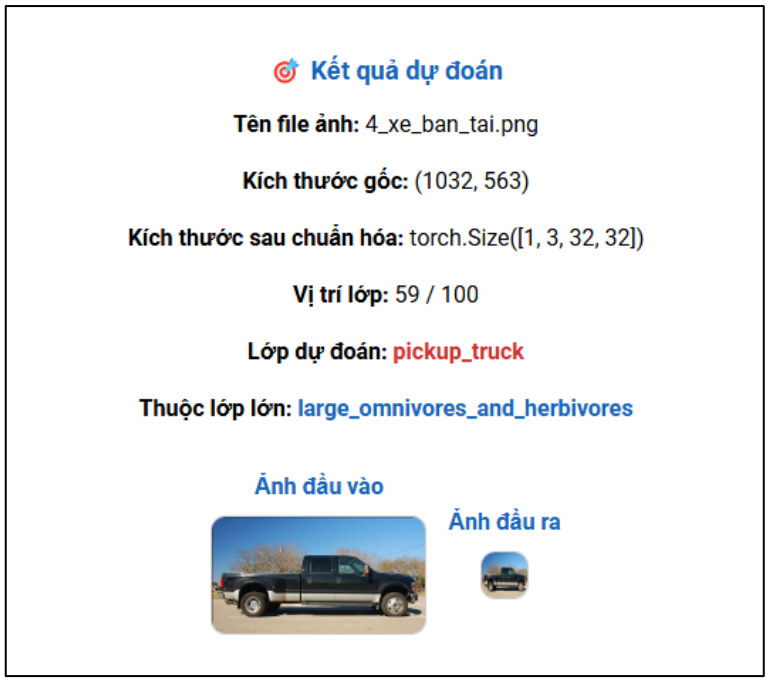
Hình 4. 2. Minh họa kết quả dự đoán là nhãn cái bàn

4.1.2. Dữ liệu thử nghiệm nhóm động vật

Hình 4. 3. Minh họa kết quả đầu ra là nhãn con cáo

Hình 4. 4. Minh họa kết quả đầu ra là nhãn con cua

4.1.3. Dữ liệu thử nghiệm nhóm phương tiện



Hình 4. 5. Minh họa kết quả đầu ra là nhãn xe bán tải

4.1.4. Dữ liệu thử nghiệm nhóm cây cối

Hình 4. 6. Minh họa kết quả đầu ra là nhãn xe đạp

4.1.5. Dữ liệu thử nghiệm nhóm thực vật

Hình 4. 7. Minh họa kết quả đầu ra là nhãn hoa lan

4.2. Cài đặt môi trường

Để bắt đầu quá trình thực thi mô hình, trước tiên cần truy cập môi trường Google Colab, nơi cung cấp các công cụ mạnh mẽ để phát triển và triển khai mô hình học sâu. Người dùng có thể truy cập vào liên kết Google Colab thông qua đường dẫn sau:

Bước 1. Truy cập vào đường dẫn Google Colab dưới đây:

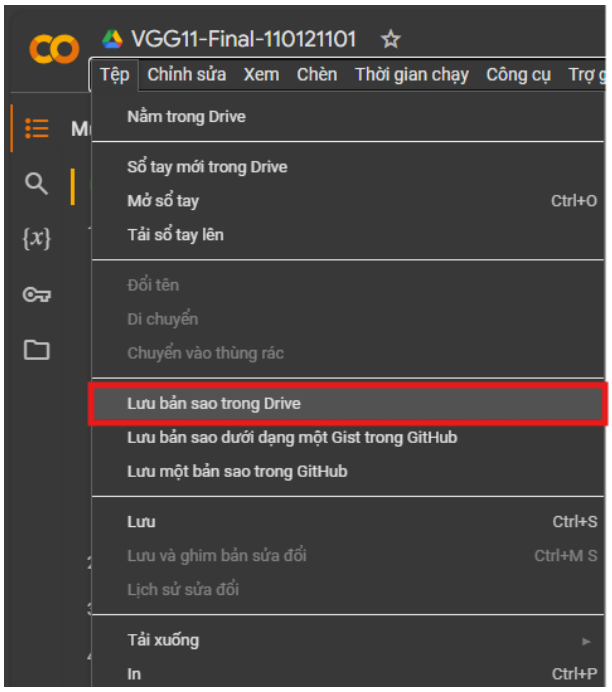
https://bom.so/iGFX5z

Bước 2. Nhấn nút **Đăng nhập:**

Hình 4. 8. Minh họa ảnh giao diện sau khi thực hiện Bước 1

Bước 3. Tiến hành đăng nhập tài khoản.

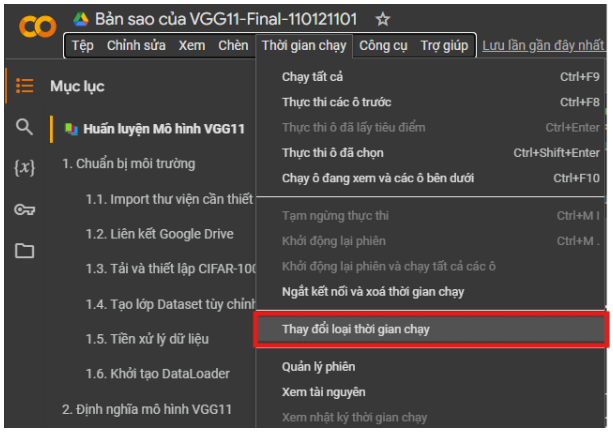
Bước 4. Ở giao diện chính chọn **Tệp** sau đó chọn **Lưu bản sao trong Drive**.



Hình 4. 9. Minh họa giao diện thao tác chọn tạo bản sao dự án

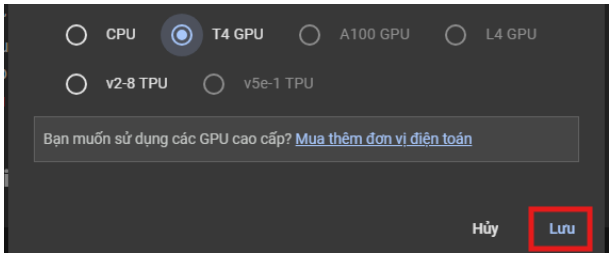
Bước 5. Sau khi hệ thống tạo bản sao xong, sẽ có hộp thoại hiện lên và chọn **Mở trong tab mới**.

Bước 6. Tại giao diện chính, chọn **Thời gian chạy** sau đó chọn **Thay đổi loại thời gian chạy**, mục đích để tiến hành thiết lập GPU.



Hình 4. 10. Minh họa giao diện thao tác thiết lập GPU

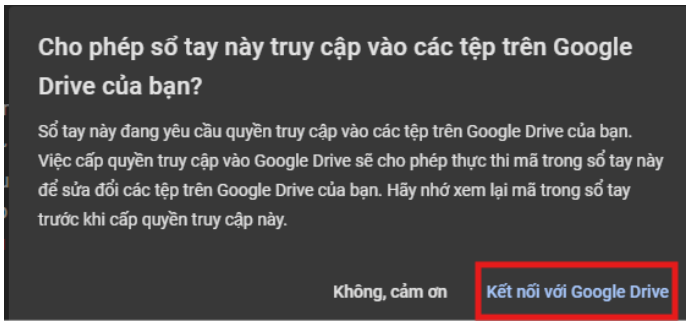
Bước 7. Lúc này sẽ có hộp thoại hiện lên và chọn **T4 GPU** sau đó chọn **Lưu**.



Hình 4. 11. Minh họa giao diện thao tác chọn T4 GPU

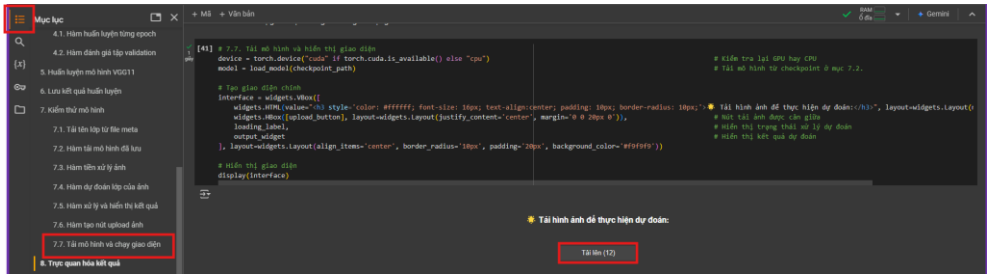
Bước 8. Nhấn tổ hợp **Ctrl + F9** để chạy toàn bộ mã có trong NoteBook hiện tại.

Bước 9. Lúc này sẽ có hộp thoại hiện lên và chọn **Kết nối Google Drive**.



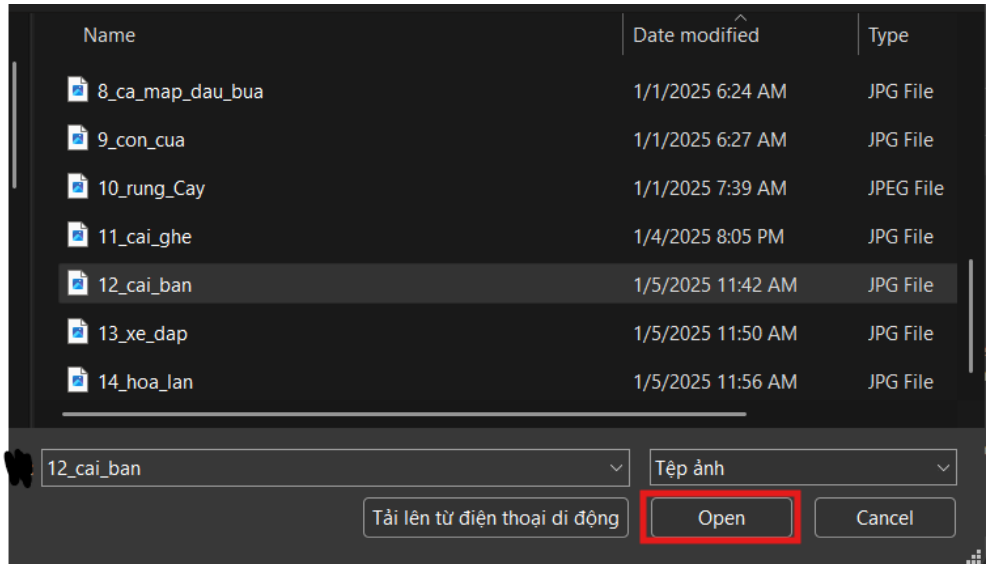
Hình 4. 12. Minh họa giao diện thao tác kết nối Google Drive

Bước 10. Sau khi thực hiện xong, lúc này người dùng cần chờ các đoạn mã chạy với thời gian dự kiến là **4 giờ 05 phút** để hoàn thành thực thi toàn bộ mã nguồn có trong NoteBook. Sau khi các đoạn mã thực thi hoàn tất, tiến hành chọn mục **7.7.** và chọn nút **Tải lên**, được minh họa như hình dưới đây để tiến hành thử nghiệm kết quả dự đoán của mô hình VGG11:



Hình 4. 13. Minh họa giao diện thao tác chuẩn bị kiểm thử mô hình VGG11

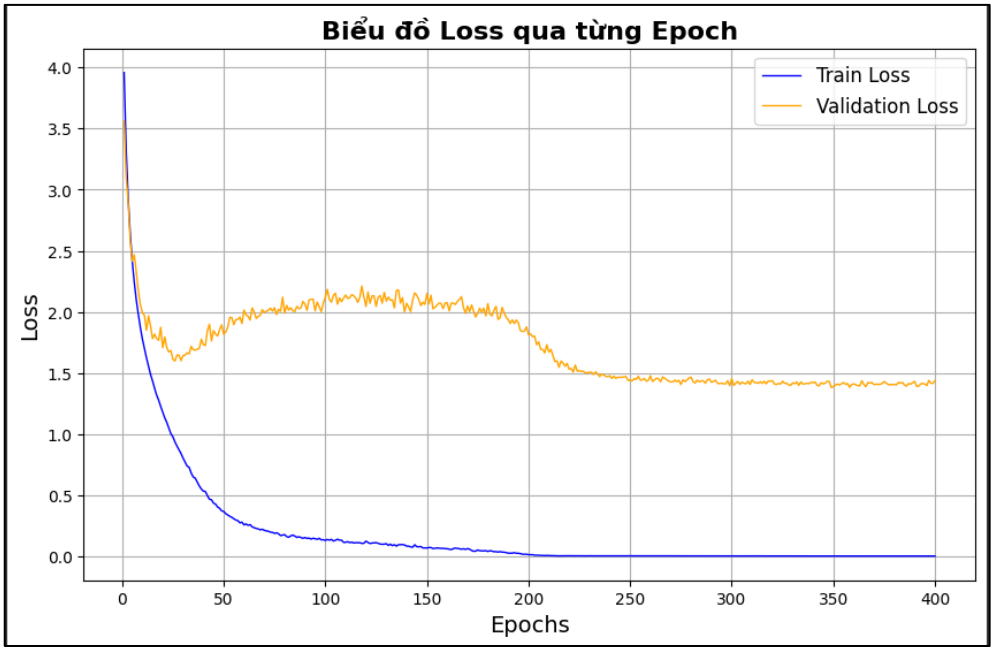
Bước 11. Tiến hành chọn ảnh muốn dự đoán và nhấn **Open**.



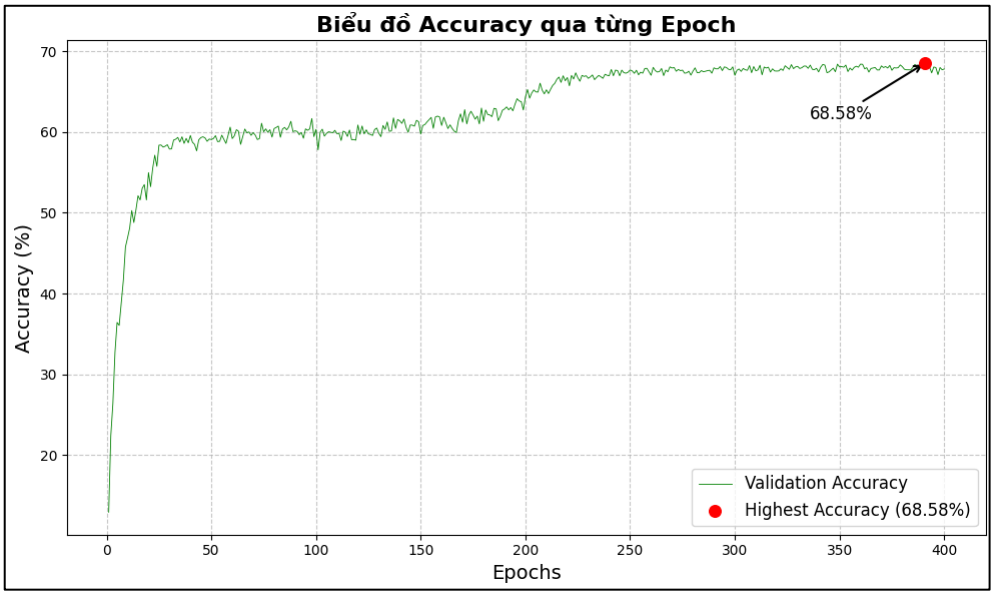
Hình 4. 14. Minh họa giao diện thao tác chọn ảnh từ máy cục bộ

Hình 4. 15. Minh họa giao diện kết quả dự đoán sau khi tải ảnh lên

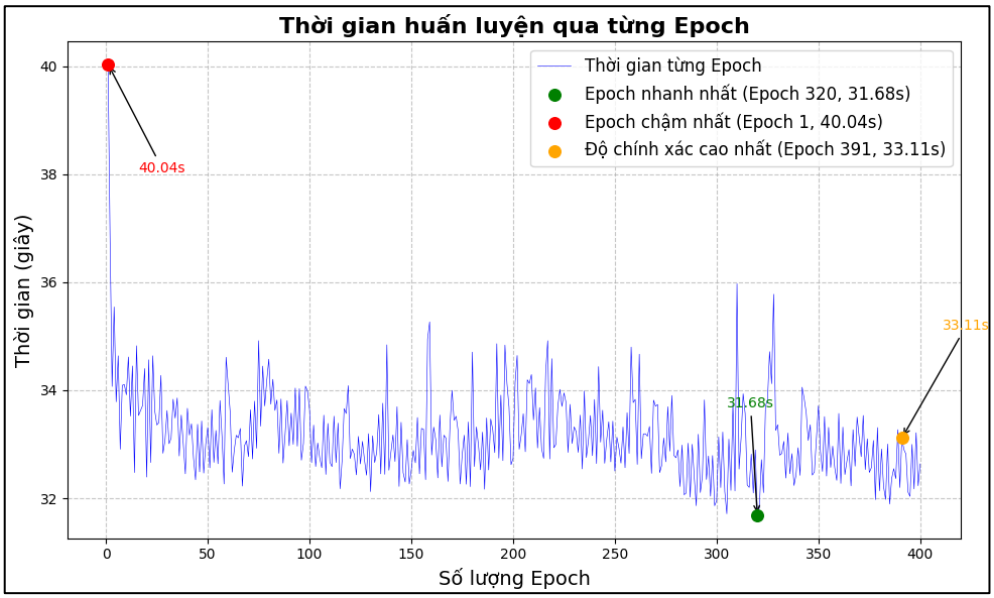
4.3. Kết quả trực quan hóa quá trình huấn luyện



Hình 4. 16. Minh họa biểu đồ huấn luyện Loss qua từng Epoch



Hình 4. 17. Minh họa biểu đồ huấn luyện Accuracy qua từng Epoch



Hình 4. 18. Minh họa biểu đồ thời gian huấn luyện qua từng Epoch

CHƯƠNG 5: KẾT LUẬN VÀ HƯỚNG PHÁT TRIỂN

5.1. Kết luận

Qua quá trình nghiên cứu và triển khai, đồ án đã chứng minh hiệu quả của mô hình VGG11 trong việc phân loại hình ảnh thuộc tập dữ liệu CIFAR-100. Từ bước chuẩn bị dữ liệu, thiết kế mô hình, đến quá trình huấn luyện và đánh giá, các mục tiêu đề ra ban đầu đều được thực hiện thành công. Mô hình không chỉ đạt độ chính xác cao trên tập kiểm tra, mà còn thể hiện khả năng học tốt các đặc trưng phức tạp của dữ liệu thông qua các kỹ thuật tối ưu hóa và tăng cường dữ liệu.

Những kết quả đạt được đã khẳng định tiềm năng ứng dụng của VGG11 trong lĩnh vực xử lý và nhận diện hình ảnh, đồng thời cho thấy rằng các kỹ thuật học sâu hiện đại có thể được tùy chỉnh và triển khai hiệu quả cho nhiều loại bài toán khác nhau. Sự kết hợp giữa kiến trúc mạnh mẽ và các phương pháp xử lý dữ liệu hợp lý đã đóng vai trò quan trọng trong thành công của nghiên cứu này.

5.2. Hạn chế

Mặc dù đạt được những kết quả tích cực, đồ án vẫn tồn tại một số hạn chế cần khắc phục:

Thời gian huấn luyện dài: Với cấu trúc mô hình lớn, việc huấn luyện trên tập dữ liệu CIFAR-100 yêu cầu thời gian dài và tài nguyên phần cứng cao.

Yêu cầu tài nguyên phần cứng lớn: Việc huấn luyện mô hình đòi hỏi sử dụng GPU, làm hạn chế khả năng triển khai trên các thiết bị không có phần cứng mạnh mẽ.

Khả năng tổng quát hóa: Mặc dù mô hình đạt hiệu quả trên CIFAR-100, nhưng cần kiểm tra thêm trên các tập dữ liệu phức tạp hoặc có độ phân giải cao hơn.

Overfitting: Một số dấu hiệu overfitting được nhận thấy trong quá trình phân tích biểu đồ huấn luyện, đặc biệt khi mô hình đạt hiệu suất cao trên tập huấn luyện nhưng chưa tương ứng trên tập kiểm tra.

5.3. Phương hướng phát triển

Dựa trên những hạn chế đã nêu, một số phương hướng phát triển cho nghiên cứu trong tương lai bao gồm:

Tối ưu hóa mô hình: Áp dụng các kỹ thuật giảm tham số như Knowledge Distillation hoặc sử dụng mô hình nhỏ gọn hơn (MobileNet, EfficientNet) để giảm thời gian huấn luyện và yêu cầu tài nguyên. Sử dụng các thuật toán tối ưu hóa hiện đại như AdamW hoặc RMSProp để cải thiện khả năng hội tụ của mô hình.

Thử nghiệm trên các tập dữ liệu lớn và phức tạp hơn: Nâng cấp bài toán từ CIFAR-100 lên các tập dữ liệu có độ phân giải cao và đa dạng hơn như ImageNet. Kiểm tra hiệu quả của mô hình trên các bài toán nhận diện đối tượng hoặc phân đoạn hình ảnh.

Tăng cường dữ liệu: Thử nghiệm các kỹ thuật tăng cường dữ liệu mới như Mixup, CutMix hoặc AugMix để tăng khả năng tổng quát hóa của mô hình.

Kết hợp các mô hình hiện đại: Kết hợp VGG11 với các kỹ thuật như Attention Mechanism (SE-Block, CBAM) hoặc thử nghiệm trên các mô hình Transformer-based như Vision Transformer (ViT) để cải thiện hiệu suất.

DANH MỤC TÀI LIỆU THAM KHẢO

[1] Visual Geometry Group, "Very Deep Convolutional Networks for Large-Scale Image Recognition," International Conference on Learning Representations (ICLR), 2015.

[2] Litjens G., et al., "A survey on deep learning in medical image analysis," Medical Image Analysis, 2017.

[3] Singh S., et al., "Object detection and face recognition for security applications using VGG-based networks," IEEE Transactions, 2018.

[4] Gatys L.A., et al., "Image Style Transfer Using Convolutional Neural Networks," CVPR, 2016.

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

[6] Kolesnikov A., et al., "Revisiting Self-Supervised Visual Representation Learning," IEEE Transactions on Pattern Analysis and Machine Intelligence, 2020.

[7] Goodfellow I., et al., "Deep Learning," MIT Press, 2016.

[8] He K., Zhang X., Ren S., Sun J., "Deep Residual Learning for Image Recognition," CVPR, 2016.

[9] Google, "Google Colaboratory: Free Jupyter Notebook Environment," Google Research, 2021.

[10] Bisong E., "Google Colaboratory," in Building Machine Learning and Deep Learning Models on Google Cloud Platform, Apress, 2019.