**A blue and black frame

Description automatically generated** BỘ CÔNG THƯƠNG

**TRƯỜNG ĐẠI HỌC CÔNG THƯƠNG TP HỒ CHÍ MINH**

**KHOA CÔNG NGHỆ THÔNG TIN**

**---------------------------**



**ĐỀ TÀI: XÂY DỰNG MÔ HÌNH DỰ ĐOÁN THỜI TIẾT**

**SINH VIÊN THỰC HIỆN**

1. 2001215850 – Huỳnh Vĩ Khang
2. 2001215964 – Đào Quí Mùi
3. 2001210540 – Võ Ngọc Hải

**GVHD: Nguyễn Thanh Long**

**BÁO CÁO GIỮA KỲ**

Môn học: Học Máy

*TP. HỒ CHÍ MINH, tháng 12 năm 2023*

Mục lục

[ĐÁNH GIÁ KẾT QUẢ LÀM VIỆC NHÓM 2](#_Toc168989287)

[LỜI CẢM ƠN 2](#_Toc168989288)

[**Chương 1:** **Mạng tích chập CNN** 2](#_Toc168989289)

[**I.** **Tích chập 2 chiều (conv2D):** 3](#_Toc168989290)

[**II.** **Đạo hàm của tích chập** 5](#_Toc168989291)

[**III.** **Một số lưu ý khi sử dụng Conv2D** 6](#_Toc168989292)

[**IV.** **Lớp liên kết đầy đủ (fully connected layer - FC)** 8](#_Toc168989293)

[**V.** **Mạng CNN** 8](#_Toc168989294)

[**VI.** **Các mạng xương sống (backbone)** 8](#_Toc168989295)

[**VII.** **VGG** 9](#_Toc168989296)

[**VIII.** **Các bài toán nhận thức hình ảnh (visual perception)** 10](#_Toc168989297)

[IX. **Kiến trúc mạng** 12](#_Toc168989298)

[**X.** **Dữ liệu** 12](#_Toc168989299)

[**Chương 2:** **CHUẨN HÓA NHÃN VÀ CHIA TÂP DỮ LIỆU** 13](#_Toc168989300)

[**1.** **Mã hóa nhãn** 13](#_Toc168989301)

[**2.** **Chia tập dữ liệu** 14](#_Toc168989302)

[**3.** **Chuân hóa dữ liệu** 14](#_Toc168989303)

[1. **Lợi Ích của Việc Xáo Trộn Dữ Liệu** 15](#_Toc168989304)

[**2.** **Huấn luyên mô hình** Các Layer Của Mô Hình CNN 16](#_Toc168989305)

[**3.** **Kiểm tra mô hình** 19](#_Toc168989306)

[**Chương 3:** **TÀI LIỆU THAM KHẢO** 25](#_Toc168989307)

ĐÁNH GIÁ KẾT QUẢ LÀM VIỆC NHÓM

|  |  |  |  |
| --- | --- | --- | --- |
| **Mã số sinh viên** | **Họ và tên** | **Nội dung công việc** | **Tiến độ** |
| 2001215850 | Huỳnh Vĩ Khang | Demo, thuyết trình thuật toán, Word | 100% |
| 2001215964 | Đào Quí Mùi | Q-Learning, PowerPoint, Word | 100% |
| 2001210540 | Võ Ngọc Hải | Double Q-Learning, PowerPoint, Word | 100% |

LỜI CẢM ƠN

Đầu tiên, chúng em xin gửi lời cảm ơn chân thành đến Trường Đại học Công Thương TP.HCM. Đặc biệt, chúng em xin gửi lời cảm ơn sâu sắc đến giảng viên bộ môn –Thầy Nguyễn Thanh Long đã dành thời gian và tâm huyết để giảng dạy cho chúng em. Những kiến thức và kinh nghiệm mà thầy đã chia sẽ giúp chúng em hiểu sâu hơn và cải thiện kỹ năng nghiên cứu của mình. Chúng em cũng muốn gửi lời cảm ơn đến các tác giả đã nghiên cứu và phát triển Ứng dựng trong trí tuệ nhân tạo và học máy. Công lao của họ đã mang lại những đóng góp to lớn cho lĩnh vực Phát triển ứng dụng trí tuệ nhân tạo.

Bộ môn Học Máy là một môn học rất thú vị và có tính ứng dụng cao. Tuy nhiên, vì kiến thức của chúng em còn hạn chế và khả năng tiếp thu thực tế còn nhiều bỡ ngỡ, nên bài tiểu luận của chúng em chắc chắn sẽ còn một số thiếu sót và chỗ chưa chính xác. Chúng em rất mong nhận được sự góp ý của thầy để hoàn thiện bài tiểu luận của nhóm mình hơn!!

1. **Mạng tích chập CNN**
2. A mathematical symbols and formulas

   Description automatically generated with medium confidenceA math equation with a white background

   Description automatically generated with medium confidence**Tích chập 2 chiều (conv2D):**

A diagram of a diagram

Description automatically generated

Kết quả (**output**) cho thấy vị trí trên ảnh đầu vào (**input**) giống với ảnh nhân (**kernel**). Ảnh nhân còn được gọi là bộ lọc (**filter**)

Ví dụ

A group of squares with numbers

Description automatically generated

Khi bộ lọc có tính đối xứng thì phép tích chập trùng với phép tương quan chéo (cross-correlation). Trong các thư viện Học máy, phép tích chập được cài đặt bằng phép tương quan chéo

A number and square symbols

Description automatically generated with medium confidence

Ví dụ: đầu vào 3 x 3, ảnh nhân 2 x 2, đầu ra 2 x 2​

A black background with blue squares and numbers

Description automatically generated

Chuyển các ảnh trên thành dạng vector được quan hệ tuyến tính

A screenshot of a computer screen

Description automatically generated

Tích chập 2 chiều ↔ Biến đổi tuyến tính với ma trận trọng số thưa có chia sẻ (số phần tử khác 0 trên mỗi dòng bằng kích thước ảnh nhân)

1. **Đạo hàm của tích chập**

A math equations on a white background

Description automatically generated

A screenshot of a computer

Description automatically generated

Các lớp mạng tích chập (CNN)

Tính toán trong Conv2D

A group of squares with numbers

Description automatically generated

Lớp Conv2D chứa bộ lọc (kích thước ) với là số kênh của đầu vào. Kích thước của tham số của lớp tích chập

* Đầu vào được nhân chập với từng bộ lọc và *cho qua hàm kích hoạt* $\phi\_l$ tạo ra các **bản đồ đặc trưng** (feature map)
* Ghép các bản đồ đặc trưng lại với nhau được ảnh đầu ra của lớp Conv2D gồm kênh
* **Ví dụ**: trong ví dụ trên, 16 nơ-ron trong bản đồ đặc trưng phía trên chia sẻ các trọng số trong bộ lọc 1, 16 nơ-ron trong bản đồ đặc trưng phía dưới chia sẻ trọng số trong bộ lọc 2.
* Bản chất lớp Conv2D vẫn là một lớp mạng trong MLP nhưng các trọng số được chia sẻ giữa các nơ-ron (làm giảm tổng số tham số). Đồng thời do tính chất của phép tích chập và kích thước bộ lọc nhỏ, kết quả tính toán thể hiện các tính chất cục bộ của ảnh đầu vào (dẫn đến tên gọi *bản đồ đặc trưng*)

1. **Một số lưu ý khi sử dụng Conv2D**

* kernel size/shape
* stride
* padding
* dilation
* Khởi tạo: ngẫu nhiên, chuẩn xavier\_initialization , uniform he\_initialization

A graph of a function

Description automatically generated

Lớp rút gọn (pooling layer - pool)

A close-up of a grid

Description automatically generated

* Làm giảm kích thước bản đồ đặc trưng —> giảm số lượng tham số / tính toán ở các lớp phía sau
* MaxPooling2D, AveragePooling2D, GlobalAveragePooling2D
* Tăng tính bất biến, tính ổn định với các biến đổi nhỏ trong đầu vào
* Kết hợp nhiều lần Conv2D, MaxPooling2D
* size, stride

1. **Lớp liên kết đầy đủ (fully connected layer - FC)**

* Đã học ở bài trước (MLP)
* Dùng để biến đổi các đặc trưng ở đoạn cuối của mạng và tạo ra lớp mạng đầu ra có số đầu ra bằng số phân lớp

1. **Mạng CNN**

* Tổ hợp các khối gồm lớp conv và lớp pool nối tiếp nhau để tính toán đặc trưng ảnh
* Các đặc trưng về sau càng có tính khái quát cao
* Sau cùng là các lớp FC để biến đổi đặc trưng và đầu ra (dùng để phân lớp, hồi quy)

A diagram of a diagram of a computer

Description automatically generated with medium confidence

1. **Các mạng xương sống (backbone)**

A graph of layers with numbers and text

Description automatically generated with medium confidence

Từ những năm 2000 đến nay, các nhà nghiên cứu đã đề xuất hàng loạt cấu trúc mạng CNN xương sống làm cơ sở cho các bài toán nhận thức hình ảnh. Mục tiêu của các mạng xương sống này là

* Có độ sâu ngày càng lớn để tính toán đặc trưng hình ảnh ngày càng khái quát và trừu tượng (abstract features)
* Có khả năng ứng dụng trong nhiều bài toán nhận thức hình ảnh khác nhau nhờ vào khái niệm trường nhìn (receptive field) của từng nơ-ron

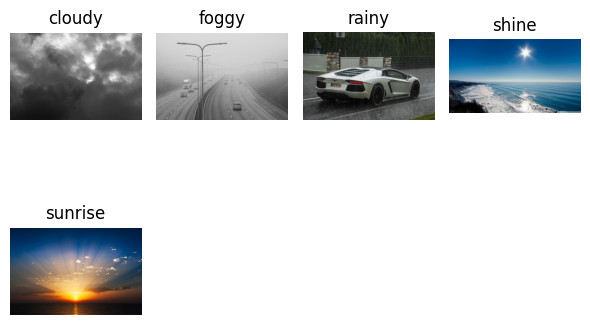
1. **VGG**

ImageNet bộ dữ liệu có 1000 phân lớpA screenshot of a computer

Description automatically generated

1. **Các bài toán nhận thức hình ảnh (visual perception)**

Bài toán nhận dạng (recognition, classification)

****

Phân loại hình ảnh là một trong những nhiệm vụ cơ bản nhất trong thị giác máy tính, nó đã tạo ra một cuộc cách mạng và thúc đẩy những tiến bộ công nghệ trong lĩnh vực AI, từ ngành công nghiệp ô tô đến phân tích y tế và nhận thức tự động trong robot.

Phân loại hình ảnh nhận đầu vào là một ảnh và gán cho nó một nhãn từ tập hợp các nhãn được xác định trước trong tập huấn luyện.

A screenshot of a screenshot of a sunset

Description automatically generated

Cách đánh giá

Dựa vào mô hình phân lớp, chia kết quả phân lớp mỗi ảnh thuộc vào các trường hợp sau: True Positive (TP), True Negative (TN), False Positive (FP), và False Negative (FN)

A table with different colored boxes

Description automatically generated with medium confidence

Dựa vào việc chia kết quả trên, có 5 thông số Accuracy, Precision, Recall and F1 score, Specificity ta tính được:

A math equations with black text

Description automatically generated with medium confidence

1. **Kiến trúc mạng**

CNN đã mang lại những kết quả đang kinh ngạc trong chục năm trở lại đây trong ngành học sâu. Rất nhiều kiến trúc mạng CNN đã được đề xuất xuyên suốt từ những năm 2012. Ban đầu, Alexnet đã đạt được độ chính xác được 63,3% Top-1 trên ImageNet. Giờ đây, kết quả đạt được trên 90% sử dụng kiến trúc EfficientNet.

A graph with a line and dots

Description automatically generated with medium confidence

1. **Dữ liệu**

Dữ liệu hình ảnh chất lượng cao để phân loại có sẵn thông qua các bộ dữ liệu công khai như ImageNet, MNIST, SVHN, CIFAR-10, CIFAR-100 và MS-COCO. Các tập dữ liệu này thường chứa hàng nghìn hình ảnh chất lượng cao, cân bằng lớp đã được chú thích và gắn nhãn chính xác.

Một trong những bộ dữ liệu phổ biến nhất cho người mới bắt đầu là bộ dữ liệu MNIST, vì nó nhỏ gọn và dễ dàng huấn luyện. Bộ dữ liệu MNIST bao gồm các chữ số viết tay, số lớp trong tập dữ liệu này là 10 được bao gồm 60000 ảnh training và 10000 ảnh testing, với mỗi ảnh có kích thước là 28x28.

Ngoài ra có rất nhiều bộ dữ liệu khác được công khai trên kaggle cho các bài toán phân lớp cụ thể, ví dụ như Dogs vs. Cats, Vehicle-classification, ...

1. **CHUẨN HÓA NHÃN VÀ CHIA TÂP DỮ LIỆU**
2. **Mã hóa nhãn**

A screenshot of a computer

Description automatically generated

1. Mã hóa one-hot (One-Hot Encoding): Đây là phương pháp phổ biến nhất. Mỗi nhãn được biểu diễn bằng một vector có độ dài bằng số lượng các nhãn có thể có trong tập dữ liệu. Tại vị trí tương ứng với nhãn, giá trị của vector sẽ là 1, còn lại là 0. Ví dụ, nếu có 3 nhãn là "A", "B", và "C", thì "A" có thể được mã hóa thành [1, 0, 0], "B" thành [0, 1, 0], và "C" thành [0, 0, 1].
2. Mã hóa nhãn số (Label Encoding): Trong trường hợp các nhãn có thứ tự, chúng ta có thể mã hóa chúng thành các giá trị số tuân thủ thứ tự đó. Ví dụ, nếu có các nhãn "Xuất sắc", "Tốt", "Trung bình", và "Yếu", chúng có thể được mã hóa thành 0, 1, 2, và 3 tương ứng.

Cả hai phương pháp đều có ưu và nhược điểm của riêng mình và sẽ được sử dụng tùy thuộc vào bản chất của dữ liệu và yêu cầu của bài toán.

1. **Chia tập dữ liệu**

Để chia tập dữ liệu thành các phần huấn luyện và kiểm tra theo tỷ lệ 80/20, bạn có thể sử dụng các hàm hoặc thư viện hỗ trợ trong các ngôn ngữ lập trình phổ biến như Python. Dưới đây là một ví dụ cách chia tập dữ liệu bằng Python, sử dụng thư viện scikit-learn:

from sklearn.model\_selection import train\_test\_split

# X là ma trận đặc trưng (features), y là vector nhãn (labels)

X\_train, X\_test, y\_train, y\_test = train\_test\_split(X, y, test\_size=0.2, random\_state=42)

Trong đoạn mã trên:

* X là ma trận đặc trưng chứa dữ liệu của bạn.
* y là vector nhãn tương ứng.
* test\_size được thiết lập thành 0.2, đại diện cho tỷ lệ 20% dữ liệu được chia cho tập kiểm tra. Điều này có nghĩa là 80% dữ liệu sẽ được chia cho tập huấn luyện.
* random\_state là một giá trị để đảm bảo rằng việc chia dữ liệu sẽ được thực hiện một cách ngẫu nhiên nhưng dự đoán. Điều này đảm bảo kết quả của bạn có thể được tái tạo nếu cần thiết.

Sau khi chia, X\_train, X\_test, y\_train, và y\_test sẽ là các biến chứa dữ liệu đã được chia thành các tập huấn luyện và kiểm tra. Bạn có thể sử dụng chúng để huấn luyện mô hình và đánh giá hiệu suất của nó.

1. **Chuân hóa dữ liệu**

* **Chia giá trị pixel cho 255:** Giá trị pixel trong ảnh thường nằm trong khoảng từ 0 đến 255, với 0 là đen và 255 là trắng. Bạn chỉ cần chia mỗi giá trị pixel cho 255 để đưa chúng về khoảng từ 0 đến 1.
* **Sử dụng thư viện phù hợp:** Trong Python, bạn có thể sử dụng thư viện PIL (Pillow) hoặc OpenCV để đọc và xử lý ảnh. Dưới đây là một ví dụ về cách thực hiện điều này sử dụng thư viện PIL:

A screenshot of a computer program

Description automatically generated

Xáo Trộn Dữ Liệu Huấn Luyện và Nhãn

Sử dụng hàm **shuffle** để xáo trộn đồng thời dữ liệu huấn luyện (**x\_train**) và nhãn tương ứng (**y\_train**). Điều này đảm bảo rằng dữ liệu và nhãn vẫn được liên kết đúng cách sau khi xáo trộn.

1. **Lợi Ích của Việc Xáo Trộn Dữ Liệu**

* **Ngăn chặn sự phụ thuộc theo thứ tự:** Xáo trộn dữ liệu giúp ngăn chặn mô hình học máy học theo một cách nào đó phụ thuộc vào thứ tự của dữ liệu, làm cho mô hình trở nên tổng quát hơn.
* **Tăng hiệu quả huấn luyện:** Xáo trộn dữ liệu thường giúp cải thiện sự hội tụ của các thuật toán học máy, đặc biệt là trong các thuật toán tối ưu hóa gradient.
* **Đảm bảo tính ngẫu nhiên:** Xáo trộn giúp tạo ra một phân phối ngẫu nhiên của dữ liệu huấn luyện, điều này có thể giúp mô hình học tốt hơn trên các mẫu dữ liệu khác nhau.

1. **Huấn luyên mô hình**Các Layer Của Mô Hình CNN

**A screen shot of a computer

Description automatically generated**

1. **Lớp Conv2D (conv2d)**:
   * **Hình dạng đầu ra**: (None, 150, 150, 32)
   * **Số lượng tham số**: 896
   * Lớp này áp dụng 32 bộ lọc tích chập kích thước 3x3 vào đầu vào.
2. **Lớp Kích hoạt (activation):**
   * Hình dạng đầu ra: (None, 150, 150, 32)
   * Số lượng tham số: 0
   * Lớp này áp dụng một hàm kích hoạt (có thể là ReLU) vào đầu ra của **lớp Conv2D trước đó.**
3. **Lớp Chuẩn hóa hàng loạt (batch\_normalization):**
   * Hình dạng đầu ra: (None, 150, 150, 32)
   * Số lượng tham số: 128
   * Lớp này chuẩn hóa đầu ra của hàm kích hoạt, ổn định và tăng tốc **quá trình huấn luyện.**
4. **Lớp Conv2D (conv2d\_1):**
   * **Hình dạng đầu ra: (None, 150, 150, 32)**
   * **Số lượng tham số: 9,248**
   * **Một lớp tích chập khác với cùng hình dạng đầu ra nhưng số lượng tham số tăng lên, chỉ ra các hoạt động bộ lọc phức tạp hơn.**
5. **Lớp Kích hoạt (activation\_1):**
   * Hình dạng đầu ra: (None, 150, 150, 32)
   * Số lượng tham số: 0
   * Áp dụng hàm kích hoạt một lần nữa.
6. **Lớp Chuẩn hóa hàng loạt (batch\_normalization\_1):**
   * Hình dạng đầu ra: (None, 150, 150, 32)
   * Số lượng tham số: 128
   * Chuẩn hóa đầu ra.
7. **Lớp Gộp tối đa (max\_pooling2d):**
   * **Hình dạng đầu ra**: (None, 75, 75, 32)
   * **Số lượng tham số**: 0
   * Giảm kích thước không gian (chiều cao và chiều rộng) xuống một nửa bằng cách sử dụng phép gộp, giữ lại các đặc trưng quan trọng trong khi giảm bớt tính toán.
8. **Lớp Conv2D (conv2d\_2):**
   * **Hình dạng đầu ra**: (None, 75, 75, 64)
   * **Số lượng tham số**: 18,496
   * Tăng độ sâu lên 64 bộ lọc, cho phép mạng học được các đặc trưng phức tạp hơn.
9. **Lớp Kích hoạt (activation\_2):**
   * **Hình dạng đầu ra**: (None, 75, 75, 64)
   * **Số lượng tham số**: 0
   * Áp dụng hàm kích hoạt.
10. **Lớp Chuẩn hóa hàng loạt (batch\_normalization\_2):**
    * **Hình dạng đầu ra**: (None, 75, 75, 64)
    * **Số lượng tham số**: 256
    * Chuẩn hóa đầu ra.
11. **Lớp Conv2D (conv2d\_3):**
    * **Hình dạng đầu ra**: (None, 75, 75, 64)
    * **Số lượng tham số**: 36,928
    * Một lớp tích chập khác với 64 bộ lọc.
12. **Lớp Kích hoạt (activation\_3):**
    * **Hình dạng đầu ra**: (None, 75, 75, 64)
    * **Số lượng tham số**: 0
    * Áp dụng hàm kích hoạt.
13. **Lớp Chuẩn hóa hàng loạt (batch\_normalization\_3):**
    * **Hình dạng đầu ra**: (None, 75, 75, 64)
    * **Số lượng tham số**: 256
    * Chuẩn hóa đầu ra.
14. **Lớp Gộp tối đa (max\_pooling2d\_1):**
    * **Hình dạng đầu ra**: (None, 37, 37, 64)
    * **Số lượng tham số**: 0
    * Tiếp tục giảm kích thước không gian.
15. **Lớp Làm phẳng (flatten):**
    * **Hình dạng đầu ra**: (None, 87616)
    * **Số lượng tham số**: 0
    * Làm phẳng đầu ra 3D thành vector 1D để chuẩn bị cho các lớp dense (kết nối đầy đủ).
16. **Lớp Dense (dense):**
    * **Hình dạng đầu ra**: (None, 512)
    * **Số lượng tham số**: 44,859,904
    * Lớp kết nối đầy đủ với 512 neuron, tăng đáng kể số lượng tham số do kết nối tất cả các kích hoạt trước đó.
17. **Lớp Kích hoạt (activation\_4):**
    * **Hình dạng đầu ra**: (None, 512)
    * **Số lượng tham số**: 0
    * Áp dụng hàm kích hoạt.
18. **Lớp Chuẩn hóa hàng loạt (batch\_normalization\_4):**
    * **Hình dạng đầu ra**: (None, 512)
    * **Số lượng tham số**: 2,048
    * Chuẩn hóa đầu ra.
19. **Lớp Dense (dense\_1):**
    * **Hình dạng đầu ra**: (None, 5)
    * **Số lượng tham số**: 2,565
    * Lớp dense cuối cùng với 5 neuron, có thể tương ứng với số lượng lớp để phân loại.
20. **Lớp Kích hoạt (activation\_5):**
    * **Hình dạng đầu ra**: (None, 5)
    * **Số lượng tham số**: 0
    * Áp dụng hàm kích hoạt cho lớp đầu ra.

Tóm tắt:

* + Mạng bao gồm nhiều lớp tích chập tiếp theo là các lớp kích hoạt và chuẩn hóa hàng loạt.
  + Các lớp gộp tối đa được sử dụng để giảm kích thước không gian và kiểm soát overfitting.
  + Lớp làm phẳng chuyển đổi các bản đồ đặc trưng 3D thành vector 1D trước khi truyền qua các lớp dense.
  + Lớp dense cuối cùng cho ra một vector 5 chiều, phù hợp cho một tác vụ phân loại với 5 lớp.
  + Mô hình có tổng cộng 44,931,609 tham số, cho thấy một kiến trúc khá phức tạp và sâu.

1. **Kiểm tra mô hình**

**A screenshot of a computer program

Description automatically generated**

**A screenshot of a computer

Description automatically generated**

Tổng Quan

* **Số lượng epoch**: 30
* **Số lượng batch mỗi epoch**: 38
* **Thời gian trung bình mỗi epoch**: Khoảng 99-145 giây

Phân Tích Chi Tiết

Epoch 1:

* **Thời gian**: 109 giây
* **Độ chính xác (accuracy)**: 0.8124
* **Mất mát (loss)**: 0.5320

Epoch 2:

* **Thời gian**: 103 giây
* **Độ chính xác**: 0.8469
* **Mất mát**: 0.4443

Epoch 3:

* **Thời gian**: 103 giây
* **Độ chính xác**: 0.8701
* **Mất mát**: 0.3275

Epoch 4:

* **Thời gian**: 99 giây
* **Độ chính xác**: 0.8995
* **Mất mát**: 0.2819

Epoch 5:

* **Thời gian**: 103 giây
* **Độ chính xác**: 0.9011
* **Mất mát**: 0.2935

Epoch 6:

* **Thời gian**: 98 giây
* **Độ chính xác**: 0.9237
* **Mất mát**: 0.1841

Epoch 7:

* **Thời gian**: 101 giây
* **Độ chính xác**: 0.9555
* **Mất mát**: 0.1475

Epoch 8:

* **Thời gian**: 100 giây
* **Độ chính xác**: 0.9693
* **Mất mát**: 0.1016

Epoch 9:

* **Thời gian**: 99 giây
* **Độ chính xác**: 0.9721
* **Mất mát**: 0.0795

Epoch 10:

* **Thời gian**: 141 giây
* **Độ chính xác**: 0.9758
* **Mất mát**: 0.0660

Epoch 11:

* **Thời gian**: 145 giây
* **Độ chính xác**: 0.9805
* **Mất mát**: 0.0615

Epoch 12:

* **Thời gian**: 142 giây
* **Độ chính xác**: 0.9895
* **Mất mát**: 0.0339

Epoch 29:

* **Thời gian**: 103 giây
* **Độ chính xác**: 0.9941
* **Mất mát**: 0.0153

Epoch 30:

* **Thời gian**: 102 giây
* **Độ chính xác**: 1.0000
* **Mất mát**: 0.0030

Nhận Xét

1. **Sự Cải Thiện Qua Các Epoch**:
   * Độ chính xác tăng đều đặn từ epoch đầu tiên (0.8124) đến epoch cuối cùng (1.0000).
   * Mất mát giảm dần từ epoch đầu tiên (0.5320) đến epoch cuối cùng (0.0030), cho thấy mô hình học rất tốt.
2. **Quá Huấn Luyện**:
   * Độ chính xác đạt mức 1.0000 ở epoch cuối cùng và mất mát rất thấp (0.0030). Điều này có thể cho thấy mô hình có thể đang bị quá huấn luyện (overfitting).
3. **Thời Gian Huấn Luyện**:
   * Thời gian huấn luyện mỗi epoch biến đổi, nhưng nhìn chung dao động khoảng từ 99 đến 145 giây.
   * Có một số epoch có thời gian dài hơn, như epoch 10 và 11.

Tổng Kết

Quá trình huấn luyện cho thấy mô hình học rất tốt, đạt độ chính xác cao và mất mát rất thấp. Tuy nhiên, cần kiểm tra kỹ hơn để đảm bảo mô hình không bị quá huấn luyện. Việc sử dụng các phương pháp kiểm tra chéo (cross-validation) hoặc tập dữ liệu kiểm thử (validation set) có thể giúp xác định điều này.

A screenshot of a graph

Description automatically generated

Ma Trận Nhầm Lẫn

Ma trận nhầm lẫn hiển thị kết quả như sau:

* **Trục hàng (true label)**: Nhãn thực tế.
* **Trục cột (predicted label)**: Nhãn được mô hình dự đoán.
* **Giá trị ô**: Số lượng mẫu và tỷ lệ phần trăm dự đoán tương ứng cho từng cặp nhãn.

Nhãn "cloudy":

* Dự đoán đúng: 54 (0.90)
* Nhầm lẫn với "foggy": 2 (0.03)
* Nhầm lẫn với "shine": 2 (0.03)
* Nhầm lẫn với "sunrise": 2 (0.03)

Nhãn "foggy":

* Dự đoán đúng: 46 (0.82)
* Nhầm lẫn với "cloudy": 1 (0.02)
* Nhầm lẫn với "rainy": 7 (0.12)
* Nhầm lẫn với "shine": 1 (0.02)
* Nhầm lẫn với "sunrise": 1 (0.02)

Nhãn "rainy":

* Dự đoán đúng: 60 (0.85)
* Nhầm lẫn với "cloudy": 3 (0.04)
* Nhầm lẫn với "foggy": 6 (0.08)
* Nhầm lẫn với "sunrise": 2 (0.03)

Nhãn "shine":

* Dự đoán đúng: 29 (0.64)
* Nhầm lẫn với "cloudy": 10 (0.22)
* Nhầm lẫn với "foggy": 3 (0.07)
* Nhầm lẫn với "rainy": 1 (0.02)
* Nhầm lẫn với "sunrise": 2 (0.04)

Nhãn "sunrise":

* Dự đoán đúng: 67 (0.99)
* Nhầm lẫn với "rainy": 1 (0.01)

Phân Tích Hiệu Suất

1. **Độ chính xác cao**:
   * Nhãn "sunrise" có độ chính xác rất cao (0.99).
   * Nhãn "cloudy" cũng có độ chính xác cao (0.90).
2. **Nhầm lẫn đáng kể**:
   * Nhãn "shine" có độ chính xác thấp nhất (0.64) và bị nhầm lẫn nhiều nhất với nhãn "cloudy" (0.22).
   * Nhãn "foggy" có một số nhầm lẫn với "rainy" (0.12).
3. **Những nhầm lẫn phổ biến**:
   * Nhãn "cloudy" và "shine" có nhầm lẫn qua lại lẫn nhau.
   * Nhãn "foggy" và "rainy" có sự nhầm lẫn đáng kể.

Tổng Kết

Mô hình hoạt động tốt với hầu hết các nhãn, đặc biệt là nhãn "sunrise" và "cloudy". Tuy nhiên, nhãn "shine" có tỷ lệ nhầm lẫn cao và cần được cải thiện. Các nhầm lẫn giữa "foggy" và "rainy" cũng đáng lưu ý và có thể yêu cầu mô hình học thêm các đặc trưng phân biệt tốt hơn giữa hai nhãn này.

Để cải thiện mô hình:

* **Tăng cường dữ liệu huấn luyện**: Bổ sung thêm mẫu dữ liệu cho các nhãn bị nhầm lẫn nhiều.
* **Tinh chỉnh mô hình**: Điều chỉnh các tham số của mô hình hoặc thử các kiến trúc khác nhau để giảm nhầm lẫn.
* **Kỹ thuật tăng cường dữ liệu**: Sử dụng kỹ thuật tăng cường dữ liệu (data augmentation) để làm phong phú thêm dữ liệu huấn luyện.

Ma trận nhầm lẫn là công cụ hữu ích để nhận diện các vấn đề cụ thể trong mô hình và giúp định hướng các cải tiến cần thiết.

Đánh giá mô hình

A comparison of a graph

Description automatically generated

Thực nhiệm trên form

A plane flying over a building

Description automatically generated

Top of Form

Bottom of Form

1. **TÀI LIỆU THAM KHẢO**

<https://towardsdatascience.com/illustrated-10-cnn-architectures-95d78ace614d>

<https://theaisummer.com/cnn-architectures/#vgg-2014>

<https://blog.exsilio.com/all/accuracy-precision-recall-f1-score-interpretation-of-performance-measures/>

<https://pytorch.org/tutorials/beginner/blitz/cifar10_tutorial.html>

<https://www.v7labs.com/blog/image-classification-guide>

https://ttsalpha.notion.site/09-10-2021-M-ng-t-ch-ch-p-CNN-483e20ea51f14bf4a5e6b5b79f950cad