BỘ GIÁO DỤC VÀ ĐÀO TẠO
ĐẠI HỌC CẦN THƠ
TRƯỜNG CÔNG NGHỆ THÔNG TIN & TRUYỀN THÔNG

BÀI TẬP LỚN 1
DEEP LEARNING

Đề tài

XÂY DỰNG MÔ HÌNH PHÂN LOẠI HÀNH ĐỘNG TRÒ CHƠI VECTOR

Sinh Viên Thực Hiện	**MSSV**
Trương Phú Quí	B2017075
Phạm Hoàng Phong	B2113316
Lê Sỹ Anh Tấn	B2113342

Cần Thơ, 10/2024

NHẬN XÉT CỦA GIẢNG VIÊN

Cần Thơ, ngày tháng năm
(Ký và ghi rõ họ tên)

LỜI CẢM ƠN

Để có được bài tập lớn số 1 này, chúng em xin được bày tỏ lòng biết ơn chân thành và sâu sắc đến thầy Phạm Nguyên Khang – người đã trực tiếp tận tình hướng dẫn,giúp đỡ em.Trong suốt quá trình thực hiện bài báo cáo này, nhờ những sự chỉ bảo và hướng dẫn quý giá đó mà bài báo cáo được hoàn thành một cách tốt nhất.

Chúng em cũng xin gửi lời cảm ơn chân thành đến các Thầy Cô Giảng viên Đại học Cần Thơ, đặc biệt là các Thầy Cô ở Khoa CNTT & TT, những người đã truyền đạt những kiến thức quý báu trong thời gian qua.

Chúng em cũng xin chân thành cảm ơn bạn bè cùng với gia đình đã luôn động viên, khích lệ và tạo điều kiện giúp đỡ trong suốt quá trình thực hiện để em có thể hoàn thành bài báo cáo một cách tốt nhất.

Tuy có nhiều cố gắng trong quá trình thực hiện, nhưng không thể tránh khỏi những sai sót. Chúng em rất mong nhận được sự đóng góp ý kiến quý báu của quý Thầy Cô và các bạn để bài báo cáo hoàn thiện hơn.

Cần Thơ, ngày tháng năm 2024

MỤC LỤC

DANH MỤC HÌNH

DANH MỤC BẢNG

PHẦN NỘI DUNG
CHƯƠNG 1 : MÔ TẢ BÀI TOÁN

1. Mô Tả Chi Tiết Bài Toán

Mục tiêu của bài toán là phát triển một mô hình phân loại hành động trong các trò chơi dựa trên dữ liệu vector hóa, sử dụng mạng nơ-ron tích chập (CNN). Trong trò chơi, các hành động của người chơi được biểu diễn dưới dạng các chuỗi vector, đại diện cho vị trí, hướng di chuyển và trạng thái khác của nhân vật trong game.

Nhiệm vụ của mô hình là phân loại chính xác hành động mà người chơi thực hiện tại mỗi thời điểm, dựa trên các thông tn được trích xuất từ hình ảnh đó.

Trong bài toán được thực hiện ở đây sẽ phân loại hành động theo 4 trạng thái:

- **Chạy** (Run): Khi **không nhấn phím** nào cả và nhân vật tự động chạy tới.
- **Nhảy** (Jump): Khi nhấn phím **Lên** sẽ nhảy qua chướng ngại vật.
- **Trượt** (Slide): Khi nhấn phím **Xuống** sẽ trượt người khỏi chướng ngại.
- **Tăng Tốc** (Sprint): Khi nhấn phím **Phải** sẽ cúi người và tăng tốc lên chạy nước rút.

2. Quy Trình Thực Hiện

- **Thu thập dữ liệu**: Thu thập hình ảnh kèm nhãn từ trò chơi, bao gồm các trạng thái hành động và nhãn phân loại tương .
- **Tiền xử lý dữ liệu**: Chuẩn hóa dữ liệu vector để đồng nhất về kích thước, phạm vi và định dạng. Quá trình này bao gồm việc **loại bỏ nhiễu**, xử lý giá trị thiếu và chuyển đổi các chuỗi vector thành dạng đầu vào phù hợp với mô hình CNN, đảm bảo tính liên tục và nhất quán của dữ liệu.
- **Xây dựng mô hình CNN**: Thiết kế kiến trúc mạng CNN bao gồm các lớp tích chập, lớp pooling, và các lớp fully connected để trích xuất đặc trưng từ dữ liệu chuỗi vector.
- **Đánh giá mô hình**: Đánh giá hiệu suất của mô hình dựa trên các chỉ số quan trọng như **Accuracy**, **Precision**, **Recall** và **F1-Score**.

CHƯƠNG 2 : THU THẬP DỮ LIỆU

2.1 Thu Thập Hình Ảnh

Để thu thập dữ liệu hình ảnh phục vụ cho việc huấn luyện mô hình, chúng em đã phát triển một chương trình tự động chụp màn hình trong quá trình chơi trò chơi thực tế. Chương trình này sẽ liên tục ghi nhận hình ảnh từ trò chơi và gán nhãn tương ứng với hành động mà người chơi thực hiện tại thời điểm đó.

Quá trình thu thập dữ liệu được thực hiện theo các bước sau:

- **Viết chương trình chụp màn hình**: Chúng em đã viết chương trình tự động chụp lại màn hình khi chơi game , màn hình sẽ được chụp lại khi:
  - Khi thực hiện bấm nút (lên, xuống, phải) thì sau một khoảng thời gian ngẫu nhiên đã được chúng em cài đặt (0.2 – 0.8 giây) sẽ tự động chụp lại màn hình gameplay và gán nhãn tương ứng là Jump, Slide, Sprint.

Hình 1. Trạng thái của các nhân vật tương ứng với hành động Jump, Sprint, Slide.

  - Sau mỗi một khoảng thời gian đã được thiết lập, cụ thể là 2 giây sẽ tự chụp màn hình và gán nhãn là Run, nếu thời điểm đó rơi vào thời gian đang delay nhấn phím thì sẽ lấy nhãn theo phím đã nhấn.

Hình 2. Trạng thái của nhân vật ở trạng thái tự động chụp (hành động Run).

- **Lưu trữ dữ liệu**: Mỗi hình ảnh được chụp lại sẽ được lưu trữ trong Folder theo nhãn. Và mỗi nhãn sẽ có khoảng 500 ảnh.
- **Kiểm tra và lọc dữ liệu**: Sau khi thu thập, chúng em tiến hành kiểm tra các hình ảnh để đảm bảo tính chính xác của nhãn và chất lượng của dữ liệu. Các hình ảnh mờ, trùng lặp hoặc không đúng với nhãn sẽ bị loại bỏ khỏi tập dữ liệu.

  Đường dẫn đến thư mục ảnh:
  https://github.com/lsanhtan/CNN_Vector/tree/main/data

2.2 Tiền Xử Lý Dữ Liệu

Sau khi hoàn tất việc thu thập dữ liệu, chúng em tiến hành các bước tiền xử lý với mục tiêu chuyển đổi các hình ảnh thô ban đầu thành dữ liệu dạng ma trận, phục vụ cho quá trình huấn luyện mô hình học máy. Quá trình tiền xử lý này là bước quan trọng nhằm đảm bảo dữ liệu có tính đồng nhất và phù hợp với yêu cầu của mô hình. Các bước cụ thể được thực hiện như sau :

- **Định dạng lại kích thước và chuẩn hóa hình ảnh**: Tất cả các hình ảnh được điều chỉnh về cùng một kích thước và tỉ lệ. Việc này giúp đảm bảo rằng mô hình có thể xử lý dữ liệu một cách nhất quán, tránh sai số khi kích thước hình ảnh không đồng đều. Ngoài ra, quá trình chuẩn hóa cũng bao gồm việc chuyển đổi giá trị điểm ảnh về phạm vi nhất định, nhằm đảm bảo các giá trị đầu vào không quá chênh lệch.
- **Chuyển đổi hình ảnh thành ma trận RGB**: Sau khi định dạng lại, mỗi hình ảnh được chuyển đổi thành ba ma trận riêng biệt, đại diện cho kênh màu **RGB** Mỗi ma trận lưu trữ thông tin về cường độ màu sắc tại từng điểm ảnh, giúp mô hình học máy có thể hiểu và phân tích hình ảnh một cách chi tiết hơn.
- **Mã hóa nhãn dưới dạng One-Hot Encoding**: Nhãn của từng hình ảnh, thay vì chỉ được biểu diễn dưới dạng một giá trị số, sẽ được chuyển đổi sang dạng One-Hot Encoding. Đây là một kỹ thuật mã hóa trong đó mỗi nhãn được biểu diễn bởi một vector nhị phân, với chỉ số tương ứng với nhãn thực sự có giá trị là 1, các giá trị còn lại là 0. Kỹ thuật này giúp mô hình dễ dàng xử lý và dự đoán chính xác các nhãn trong quá trình huấn luyện."

CHƯƠNG 3 : XÂY DỰNG VÀ ĐÁNH GIÁ MÔ HÌNH CNN

3.1 Mô Hình CNN Thứ 1

3.1.1 Kiến Trúc Mô Hình

Mô hình thử nghiệm đầu tiên được xây dựng với kiến trúc như sau:

- Ảnh đầu vào có kích thước là **224x224** với 3 kênh màu (**RGB**).
- Mô hình gồm **3** lớp tích chập (**Conv2D**) với lần lượt là **32**, **64** và **128** filters có kích thước cố định là **3x3** và không thực hiện **Padding**, **Stride** hay **Dilation**. Hàm kích hoạt sử dụng là **ReLU**.
- Sau mỗi lớp tích chập sẽ là một lớp giảm chiều **MaxPooling2D** với kích thước **pool size** là **2x2** giúp làm giảm đi một nửa kích thước của bức ảnh.
- Lớp **Flatten** để chuyển dữ liệu về dạng vector phù hợp để làm đầu vào cho các lớp Dense ở phía sau.
- Lớp **Dropout** với tỉ lệ 50% để vô hiệu hoá ngẫu nhiên một nửa neurons.
- Lớp **Dense** đầu tiên có **128 Neurons**, thực hiện **Fully Connected** với tất cả các lớp đầu vào trước đó. Dùng hàm kích hoạt là **ReLU**.
- Lớp **Dense** cuối cùng với **4 Neurons** tương ứng với **4 lớp đầu ra** là dự đoán của mô hình, hàm kích hoạt sử dụng là **Softmax**.

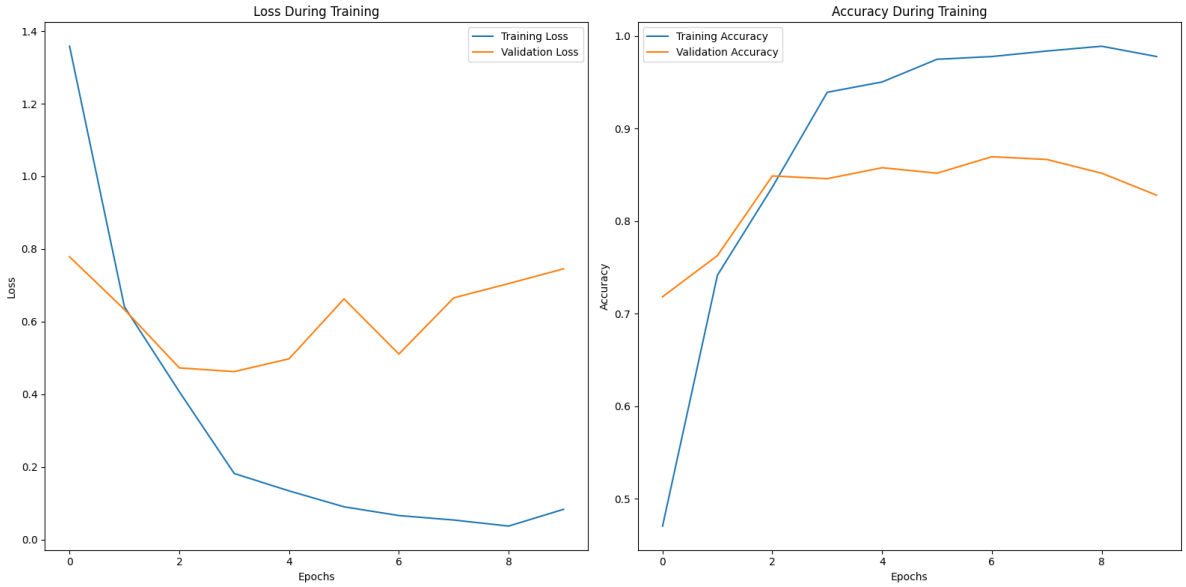
Layer	Output Shape	Params #
Conv2D	(None, 222, 222, 32)	896
MaxPooling2D	(None, 111, 111, 32)	0
Conv2D_1	(None, 109, 109, 64)	18.496
MaxPooling2D_1	(None, 54, 54, 64)	0
Conv2D_2	(None, 52, 52, 128)	73.856
MaxPooling2D_2	(None, 26, 26, 128)	0
Flatten	(None, 86528)	0
Dropout	(None, 86528)	0
Dense	(None, 128)	11.075.712
Dense_1	(None, 4)	516

Bảng 1. Kiến trúc mô hình CNN đầu tiên

- Total Params: 11.169.476 (42.61 MB).
- Trainable Params: 11.169.476 (42.61 MB).
- Non-trainable Params: 0 (0 MB).

3.1.2 Kết Quả Thu Được

- Tập dữ liệu được chia thành các tập **Train/Test** với tỷ lệ **8/10** và **2/10**.
- Tiếp tục chia dữ liệu **Train/Validation** với tỉ lệ **8/10** và **2/10**
- Huấn luyện mô hình với **batch size** = 32 và **epochs** = 10, kết quả thu được của loss trên tập train và validate như sau:



Hình 3. Biểu đồ tương quan của chỉ số Loss và Accuracy của mô hình 1

- Các chỉ số đánh giá thu được:

Accuracy	Precision	Recall	F1 Score
82.19%	82.21%	82.19%	81.9%

Bảng 2. Các chỉ số đánh giá thu được của mô hình 1

3.1.3 Nhận Xét Mô Hình

- Loss trên tập Train giảm đều cho thấy mô hình đang học tốt trên tập dữ liệu huấn luyện hiện có.
- Tuy nhiên, Loss trên tập Validation ban đầu giảm nhẹ nhưng sau đó có sự dao động và không còn xu hướng giảm (từ epoch 4 trở đi). Dẫn đến sự chênh lệch khá lớn khi so với Loss trên tập Train sau khi huấn luyện.
- **Overfit:** Từ đồ thị có thể kết luận được mô hình bị **overfit** chỉ sau một vài epoch đầu tiên. Điều này xảy ra có thể do mô hình hiện tại đang quá phức tạp với rất nhiều tham số, số lượng dữ liệu chưa đủ nhiều và còn bị nhiễu.

3.2 Mô Hình CNN Thứ 2

Ở mô hình thứ 2, chúng em nhận thấy được từ mô hình đầu tiên là có quá nhiều tham số tuy nhiên kết quả lại có xu hướng overfit nên chúng em đã tiến hành thay đổi lại kích thước đầu vào của ảnh từ **224x224** sang **64x64**, đồng thời giảm bớt độ phức tạp của mô hình (giảm số lượng tham số).

3.2.1 Cấu Trúc Mô Hình

- Sử dụng **2** lớp tích chập **Conv2D** với số lượng filters lần lượt là 16 và 32 cho từng lớp, cả 2 lớp đều sử dụng bộ lọc có kích thước 3x3 và không thực hiện **Padding**, **Stride** hay **Dilation**.
- Sau mỗi lớp tích chập là lớp giảm chiều **MaxPooling2D** với kích thước **2x2** làm giảm một nửa kích thước của ảnh.
- Sử dụng một lớp **Flatten** để chuyển đầu ra của những lớp trước về thành dạng vector phù hợp cho lớp Dense phía sau.
- Lớp **Dropout** với tỉ lệ 50% để vô hiệu hoá ngẫu nhiên một nửa Neuron.
- Lớp **Dense** với **32 Neuron** thực hiện **Fully Connected** và sử dụng hàm kích hoạt **ReLU**.
- Lớp **Dense** cuối cùng với **4 Neuron** tương ứng với **4 lớp đầu ra** là dự đoán của mô hình, hàm kích hoạt sử dụng là **Softmax**.

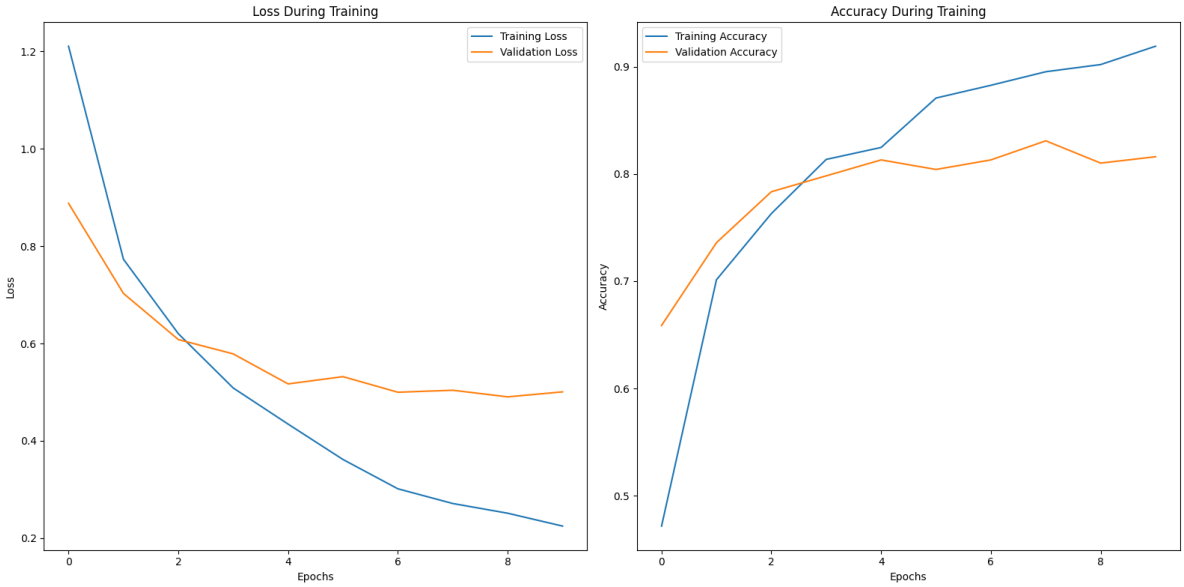
Layer	Output Shape	Pararm #
Conv2D	(None, 62, 62, 16)	448
MaxPooling2D	(None, 31, 31, 16)	0
Conv2D_1	(None, 29, 29, 32)	4.640
MaxPooling2D_1	(None, 14, 14, 32)	0
Flatten	(None, 6272)	0
Dropout	(None, 6272)	0
Dense	(None, 32)	200.736
Dense_1	(None, 4)	132

Bảng 3. Kiến trúc mô hình CNN thứ 2

- Total Params: 205.956 (804.52 KB)
- Trainable Params: 205.956 (854.52 KB)
- Non-trainable Params: 0 (0 MB)

3.2.2 Kết Quả Thu Được

- Chia dữ liệu train/test với tỉ lệ 8/10 và 2/10
- Tiếp tục chia dữ liệu train/validation với tỉ lệ 8/10 và 2/10
- Thực hiện việc Huấn luyện mô hình với 10 Epoch và Batch Size là 32



Hình 4. Biểu đồ tương quan của chỉ số Loss và Accuracy của mô hình 2

Với các chỉ số đánh giá lần được sau đây :

Accuracy	Precision	Recall	F1 Score
82.66%	83.49%	82.66%	82.82%

Bảng 4. Các chỉ số đánh giá thu được của mô hình 2

3.2.3 Nhận Xét Mô Hình

- **Hiệu suất ban đầu tốt**: Cả training và validation loss đều giảm đáng kể sau vài epoch đầu tiên, cho thấy mô hình đang học tốt các đặc trưng từ dữ liệu.
- **Overfitting nhẹ**: Sau khoảng epoch thứ 2, validation loss không tiếp tục giảm đều mà dao động, trong khi training loss tiếp tục giảm. Điều này có thể là dấu hiệu của việc mô hình dần dần quá khớp với tập huấn luyện.

3.3 Mô Hình CNN Thứ 3

Quan sát thấy vấn đề overfitting đã giảm đối với việc cải thiện của model 2 và khả năng có thể cải thiện thêm nên chúng em vẫn giữ kích thước ảnh là **64x64** và **tăng thêm một lớp tích chập** nhằm để mô hình học được các đặc trưng chi tiết hơn từ ảnh.

3.3.1 Cấu Trúc Mô Hình

Mô hình thử nghiệm thứ ba được xây dựng với kiến trúc tương tự như mô hình 2, bổ sung thêm 1 tầng gồm **Conv2D** và **Maxpooling2D** để làm giảm tham số:

- **3** lớp tích chập (**Conv2D**) với số lượng filters lần lượt là **16, 32, 64** và mỗi filter có kích thước **3x3**, không thực hiện **Padding**, **Stride**, **Dilation**.
- **3** lớp giảm chiều **MaxPooling2D** theo sau các lớp tích chập với kích thước **2x2** làm giảm đi một nửa kích cỡ ảnh.
- Sử dụng một lớp **Flatten** để chuyển đổi thành vector cho quá trình xử lý kế tiếp ở lớp Dense.
- Lớp **Dropout** với tỉ lệ 50% để tắt ngẫu nhiên một nửa số nơ ron nhằm cải thiện vấn đề overfitting.
- Lớp **Dense** với **64 Neurons** sử dụng hàm kích hoạt **ReLU** thực hiện **Fully Connected**.
- Lớp **Dense** đầu ra với với **4 neurons** và sử dụng hàm kích hoạt là **Softmax**.

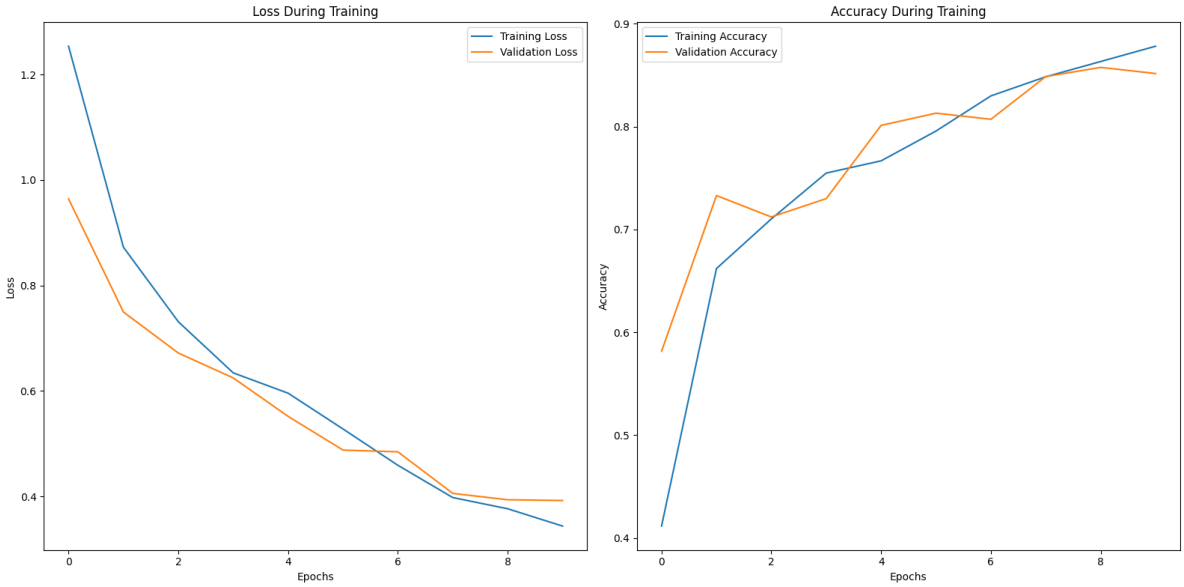
Layer	Output Shape	Pararm #
Conv2D	(None, 62, 62, 16)	448
MaxPooling2D	(None, 31, 31, 16)	0
Conv2D_1	(None, 29, 29, 32)	4.640
MaxPooling2D_1	(None, 14, 14, 32)	0
Conv2D_2	(None, 12, 12, 64)	18.496
MaxPooling2D_2	(None, 6, 6, 64)	0
Flatten	(None, 2304)	0
Dropout	(None, 2304)	0
Dense	(None, 64)	147.520
Dense_1	(None, 4)	260

Bảng 5. Kiến trúc mô hình CNN thứ 3

- Total Params: 171.364 (669.39 KB)
- Trainable Params: 171.364 (669.39 KB)
- Non-trainable Params: 0 (0 MB)

3.3.2 Kết Quả Thu Được

- Dữ liệu được chia thành 2 tập train/test với tỉ lệ 8/10 và 2/10.
- Tiến hành chia dữ liệu train và validation với tỉ lệ 8/10 và 2/10
- Thực hiện việc Huấn luyện mô hình với 10 Epoch và Batch Size là 32



Hình 5. Biểu đồ tương quan của chỉ số Loss và Accuracy của mô hình 3

- Tổng hợp kết quả các chỉ số đánh giá:

Accuracy	Precision	Recall	F1 Score
81.24%	81.1%	81.24%	80.62%

Bảng 6. Các chỉ số đánh giá thu được của mô hình 3

3.3.3 Nhận Xét Mô Hình

- Kết quả dự đoán được cải thiện sau khi thực hiện thêm vào một lớp tích chập cho mô hình.
- Chỉ số Loss trên tập validate giảm dần theo từng epoch và không chênh lệch quá nhiều so với tập train.
- Mô hình không bị overfitting do có quá trình học tốt.

KẾT LUẬN

1. Kết Quả Đạt Được

- Mô hình đầu tiên tuy có kết quả tương đối khả quan, nhưng số lượng tham số lớn dẫn đến hiện tượng **overfitting**, khiến mô hình không tổng quát hóa tốt trên tập kiểm thử.
- Sau khi giảm kích thước đầu vào và đơn giản hóa mô hình, mô hình thứ hai đã **giảm** được **số lượng tham số**, tuy nhiên vẫn gặp tình trạng **overfitting nhẹ**.
- Ở mô hình thứ ba, chúng em đã tối ưu hơn với sự kết hợp giữa số lượng lớp Conv2D và MaxPooling phù hợp, giúp mô hình học tốt và giảm thiểu hiện tượng overfitting. Kết quả đạt được trên tập kiểm thử cho thấy độ chính xác cao và không có sự chênh lệch lớn giữa tập huấn luyện và kiểm thử.

2. Hướng Phát Triển

- **Mở rộng dữ liệu**: Một trong những cách để cải thiện mô hình là tiếp tục thu thập thêm dữ liệu và áp dụng các kỹ thuật data augmentation để tăng tính đa dạng cho tập huấn luyện.
- **Tối ưu hóa mô hình**: Có thể thử nghiệm thêm các kiến trúc mạng sâu hơn, hoặc điều chỉnh các hyperparameter như số lượng epochs, batch size, và learning rate để tối ưu quá trình học của mô hình.
- **Sử dụng các mô hình tiên tiến**: Thử áp dụng các mô hình mạng nơ-ron tiên tiến hơn như ResNet hay EfficientNet để cải thiện khả năng trích xuất đặc trưng và nâng cao độ chính xác của mô hình phân loại hành động.
- **Tối ưu hóa quá trình huấn luyện**: Áp dụng các kỹ thuật như early stopping và regularization để ngăn chặn hiện tượng overfitting và tối ưu hóa hiệu suất mô hình.

TÀI LIỆU THAM KHẢO

[1]. Slide bài giảng học phần Deep Learning – Thầy Phạm Nguyên Khang.