BỘ GIÁO DỤC VÀ ĐÀO TẠO

TRƯỜNG ĐẠI HỌC SƯ PHẠM KỸ THUẬT
KHOA CÔNG NGHỆ THÔNG TIN

CONVOLUTIONAL NEURAL NETWORKS (CNNS) & DEEPFAKE DETECTION

ĐỒ ÁN CÔNG NGHỆ THÔNG TIN

MÃ MÔN HỌC : PROJ215879_12CLC

NHÓM THỰC HIỆN: 03_CLC_TV

THÀNH VIÊN: NGUYỄN ĐỨC THỊNH – 23110156

GIÁO VIÊN HƯỚNG DẪN: TS. LÊ VĂN VINH

Tp.Hồ Chí Minh, tháng 10 năm 2025

NHẬN XÉT CỦA GIẢNG VIÊN

Họ và Tên sinh viên: Nguyễn Đức Thịnh MSSV: 23110156

Ngành: Công nghệ Thông tin

Tên đề tài: Convolutional Neural Networks (CNNs) & Deepfake detection

Họ và tên Giảng viên hướng dẫn: TS. Lê Văn Vinh

NHẬN XÉT:

1. Về nội dung đề tài khối lượng thực hiện:

……

……

2. Ưu điểm:

……

……

3. Khuyết điểm:

……

……

4. Điểm :

……

……

Tp. *Hồ Chí Minh, ngày tháng năm 2022*

Giảng viên hướng dẫn

(Ký & ghi rõ họ tên)

MỤC LỤC

PHẦN MỞ ĐẦU

1. Lý do chọn đề tài

Tính cấp thiết của xã hội: Sự phát triển vượt bậc của các mô hình Generative AI (GANs, Diffusion Models, Flow Matching) khiến việc tạo ra hình ảnh và video giả mạo (Deepfake) trở nên dễ dàng và chân thực đến mức mắt thường khó phân biệt. Điều này gây ra những rủi ro nghiêm trọng về an ninh thông tin, lừa đảo tài chính và bôi nhọ danh dự.

Khoảng trống kỹ thuật: Các phương pháp xác thực sinh trắc học truyền thống đang dần trở nên vô hiệu trước Deepfake thế hệ mới. Cần có các công cụ tự động để hỗ trợ con người.

Về CNN: CNNs là kiến trúc nền tảng cơ bản trong Computer Vision. CNN có khả năng tự động trích xuất các đặc trưng không gian từ mức thấp (cạnh, góc) đến mức cao (kết cấu da, mắt), giúp phát hiện các dấu hiệu giả mạo tinh vi mà con người bỏ sót.

2. Mục tiêu nghiên cứu

Mục tiêu chung:	Xây dựng và tối ưu hóa mô hình Deep Learning dựa trên kiến trúc CNN để phân loại chính xác hình ảnh/video là thật hay giả.
Mục tiêu cụ thể:	- Tìm hiểu và làm mẫu một mô hình CNN. - Tìm hiểu và áp dụng thêm các mô hình CNN có sẵn bằng transfer learning để giúp mô hình mạnh mẽ hơn. - Đánh giá hiệu suất mô hình trên các bộ dữ liệu chuẩn (FaceForensics++, Celeb-DF) với các chỉ số như Accuracy, Precision, Recall và AUC.

3. Phương pháp nghiên cứu

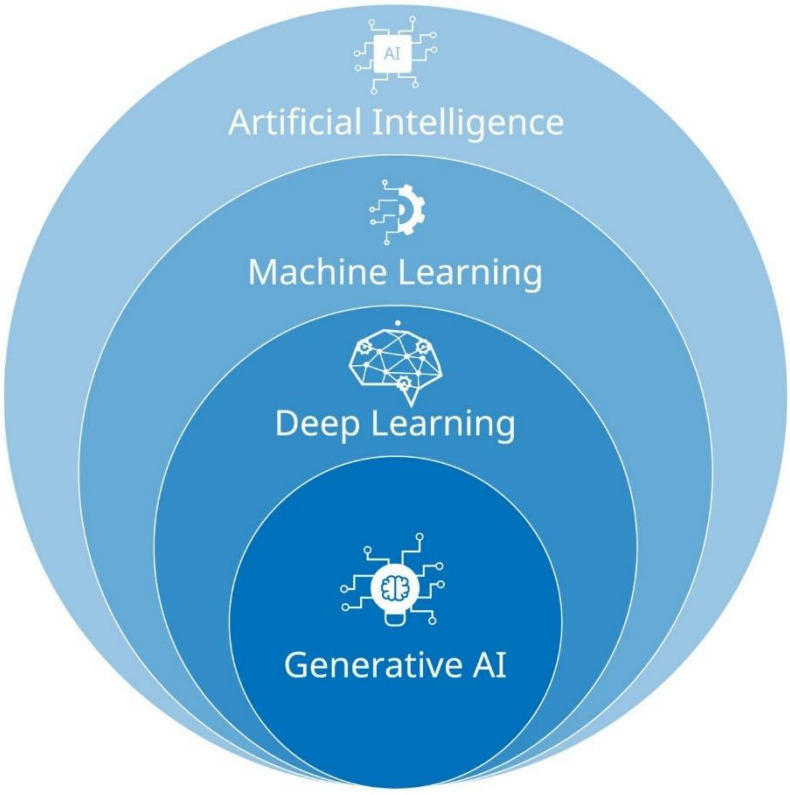
Data Collection Method	- Sử dụng các bộ dữ liệu mã nguồn mở uy tín (Dataset) chuyên về Deepfake. - Phân chia dữ liệu theo tỷ lệ chuẩn.
Experimental Method	- **Preprocessing:** Chuẩn hóa kích thước ảnh. - **Modeling:** Huấn luyện mô hình CNN theo phương pháp học có giám sát. - **Evaluation:** So sánh kết quả thực nghiệm giữa các kiến trúc khác nhau.

4. Kỹ thuật nghiên cứu

Research Environment	Google Colab, Kaggle
Data Processing	Resize, normalize
Deep Learning	CNN: convolution, pooling, FC layers Activation: ReLU, LeakyReLU Regularization: dropout, batchnorm Transfer Learning: MobileNetV2
Optimization	Optimizers: AdamW, SGD Learning rate scheduling

PHẦN NỘI DUNG

CHƯƠNG 1: TỔNG QUAN VỀ CNN



1.1. Artificial Intelligence (AI)

AI là lĩnh vực nghiên cứu phát triển các hệ thống máy tính có khả năng mô phỏng hành vi thông minh của con người như: học hỏi, suy luận, nhận thức hình ảnh, xử lý ngôn ngữ tự nhiên, ra quyết định và thích nghi với môi trường.

AI đã phát triển mạnh mẽ nhờ sự kết hợp của ba yếu tố:

1. Lượng dữ liệu khổng lồ (Big Data)
2. Sức mạnh tính toán tăng cao (GPU, TPU)
3. Thuật toán hiện đại (Machine Learning, Deep Learning)

Ứng dụng AI xuất hiện trong: xe tự hành, nhận dạng giọng nói, chatbot, camera thông minh, y tế, tài chính…

Trong ngữ cảnh của đề tài này, chủ yếu tập trung vào Narrow AI, hay còn gọi là AI yếu. Khác với AI tổng quát, Narrow AI được thiết kế để thực hiện một nhiệm vụ cụ thể với hiệu suất vượt trội. Hệ thống Deepfake Dêtction là một ví dụ điển hình của Narrow AI, được huấn luyện chuyên biệt để phân biệt thật/giả dựa trên tập dataset chuyên biệt.

1.2. Machine Learning (ML)

ML là một nhánh của AI tập trung vào việc xây dựng thuật toán cho phép máy tính học từ dữ liệu và đưa ra dự đoán mà không cần lập trình cứng.

ML gồm có ba nhóm chính:

Supervised Learning (Học có giám sát)	Dữ liệu có nhãn → dùng để phân loại ảnh, nhận diện khuôn mặt thật/giả.
Unsupervised Learning (Học không giám sát)	Tìm cấu trúc ẩn: phân cụm khách hàng, giảm chiều dữ liệu.
Reinforcement Learning (Học tăng cường)	Agent tự học thông qua tương tác môi trường: chơi game, robot.

Tuy nhiên, ML truyền thống (như SVM, Random Forest) gặp hạn chế lớn trong bài toán xử lý ảnh: quá trình trích xuất đặc trưng (feature extraction) phải được thực hiện thủ công bởi con người (hand-crafted features). Điều này tốn kém thời gian và thường bỏ sót các đặc trưng phức tạp của Deepfake.

1.3. Deep Learning (DL)

DL là một bước tiến hóa của ML, lấy cảm hứng từ cấu trúc và chức năng của bộ não con người, được gọi là Mạng nơ-ron nhân tạo (Artificial Neural Networks - ANN). Điểm khác biệt mang tính cách mạng của DL so với ML truyền thống là khả năng tự động trích xuất đặc trưng (Automatic Feature Extraction).

- Trong DL, dữ liệu đi qua nhiều lớp xử lý.
- Các lớp đầu tiên học các đặc trưng đơn giản (cạnh, góc).

- Các lớp sâu hơn học các đặc trưng phức tạp và trừu tượng hơn.

Nhờ kiến trúc sâu (nhiều lớp ẩn), DL xử lý cực tốt các dữ liệu phi cấu trúc như hình ảnh, âm thanh và văn bản, làm nền tảng cho sự ra đời của các mô hình phát hiện Deepfake hiện đại.

1.4. Convolutional Neural Networks

CNNs là mô hình DL phổ biến nhất cho xử lý ảnh, ra đời năm 1998 (LeNet) và phát triển mạnh từ AlexNet (2012).

CNN được thiết kế dựa trên 3 đặc tính quan trọng của ảnh:

1. **Tính cục bộ** (Local Connectivity)

   Mỗi kernel chỉ "nhìn" một vùng nhỏ của ảnh → trích xuất đặc trưng như cạnh, đường cong, texture.

2. **Tính chia sẻ trọng số** (Weight Sharing)

   Một kernel được áp dụng cho toàn bộ ảnh → giảm số lượng tham số, tăng hiệu quả học.

3. **Tính phân cấp đặc trưng** (Hierarchical Features)

   CNN học đặc trưng từ thấp → cao:

   Layer đầu: cạnh, góc

   Layer giữa: hình dạng

   Layer cuối: khuôn mặt, vật thể

1.4.1. Lớp tích chập (Convolutional Layer)

Đây là thành phần cốt lõi của CNN. Lớp này sử dụng các bộ lọc (filters/kernels) trượt qua toàn bộ bức ảnh đầu vào để thực hiện phép toán tích chập.

Mục tiêu: Tạo ra các bản đồ đặc trưng (Feature Maps).

Cơ chế: Các bộ lọc giúp phát hiện các mẫu cục bộ như đường thẳng, đường cong, hoặc các kiến trúc phức tạp hơn ở các lớp sâu. Trong bài toán Deepfake, lớp này giúp phát hiện các điểm bất thường về pixel hoặc noise.

1.4.2. Lớp gộp (Pooling Layer)

Thường được đặt sau lớp tích chập (phổ biến nhất là Max Pooling).

Mục tiêu: Giảm kích thước không gian của dữ liệu (giảm chiều) để giảm khối lượng tính toán và tránh hiện tượng quá khớp (overfitting).

Ý nghĩa: Giúp mô hình tập trung vào các đặc trưng quan trọng nhất và tạo ra tính bất biến (invariance) – tức là dù khuôn mặt có dịch chuyển nhẹ trong khung hình, mô hình vẫn nhận diện được.

1.4.3. Lớp kết nối đầy đủ (Fully Connected Layer)

Nằm ở cuối mạng CNN. Sau khi ảnh đi qua các lớp Convolution và Pooling, dữ liệu được duỗi phẳng (flatten) thành vector và đưa vào lớp FC.

Mục tiêu: Phân loại.

Cơ chế: Lớp này hoạt động như một mạng nơ-ron truyền thống, kết hợp tất cả các đặc trưng đã học được để đưa ra xác suất cuối cùng: Real/Fake

CHƯƠNG 2: CÁC KIẾN TRÚC CNN PHỔ BIẾN & KỸ THUẬT TRANSFER LEARNING

2.1. Cơ chế hoạt động chuyên sâu của CNN

2.1.1. Các tham số trong lớp tích chập

Quá trình tích chập được điều khiển bởi các siêu tham số (hyperparameters) quan trọng:

- **Kernel Size (Kích thước bộ lọc):** Thường là 3x3, 5x5 hoặc 7x7. Bộ lọc nhỏ (3x3) thường được ưa chuộng trong các mạng sâu (như VGG, ResNet) vì giảm chi phí tính toán nhưng vẫn giữ được khả năng bắt các chi tiết nhỏ (nhiễu pixel trong Deepfake).
- **Stride (Bước trượt):** Là số pixel mà bộ lọc dịch chuyển mỗi lần. S = 1 giữ nguyên độ phân giải không gian, trong khi S = 2 làm giảm kích thước ảnh đi một nửa (tương tự pooling).
- **Padding (Lề):** Thêm các pixel giá trị bằng 0 bao quanh ảnh đầu vào.
    - *Valid Padding:* Không thêm lề, kích thước ảnh giảm sau mỗi lớp.
    - *Same Padding:* Giữ nguyên kích thước ảnh đầu ra so với đầu vào, quan trọng để xây dựng các mạng nơ-ron sâu (Deep Neural Networks).

2.1.2. Hàm kích hoạt phi tuyến (Non-linear Activation Functions)

Nếu chỉ có các phép nhân cộng tuyến tính, CNN không thể học được các mẫu phức tạp. Hàm kích hoạt giúp mô hình giải quyết các biên quyết định phi tuyến.

- **ReLU:** $f(x) = \max(0,x)$
    - Là hàm phổ biến nhất hiện nay. Giúp mạng hội tụ nhanh và giảm thiểu vấn đề biến mất đạo hàm (vanishing gradient).
- **Softmax:** Thường dùng ở lớp cuối cùng để chuyển đổi vector đặc trưng thành xác suất có giá trị từ 0 đến 1.

2.1.3. Batch Normalization và Dropout

- **Batch Normalization:** Chuẩn hóa đầu ra của lớp trước đó về phân phối chuẩn. Giúp ổn định quá trình huấn luyện và cho phép dùng learning rate cao hơn.

- **Dropout:** Ngẫu nhiên "tắt" một số nơ-ron trong quá trình huấn luyện. Kỹ thuật này cực kỳ quan trọng trong Deepfake detection để ngăn mô hình "học vẹt" (overfitting) vào một bộ dữ liệu cụ thể.

2.2. Kỹ thuật Transfer Learning (Học chuyển tiếp)

2.2.1. Khái niệm

Transfer Learning là kỹ thuật sử dụng một mô hình đã được huấn luyện trước trên một tập dữ liệu khổng lồ (thường là **ImageNet** với 14 triệu ảnh) để giải quyết một bài toán mới tương tự.

Thay vì khởi tạo trọng số ngẫu nhiên (random initialization), ta tận dụng các "kiến thức" (trọng số) mà mạng đã học được (như cách nhận diện cạnh, góc, texture) để áp dụng sang bài toán phân biệt thật/giả.

2.2.2. Các chiến lược Transfer Learning

Có hai cách tiếp cận chính khi áp dụng vào Deepfake detection:

1. **Feature Extraction (Trích xuất đặc trưng):**

   Freeze toàn bộ các lớp Convolutional (giữ nguyên trọng số ImageNet). Chỉ thay thế và huấn luyện lại các lớp Fully Connected ở cuối.

   Ưu điểm: Nhanh, tốn ít tài nguyên.

2. **Fine-tuning (Tinh chỉnh):**

   Sau khi thay thế lớp cuối, ta Unfreeze một số hoặc tất cả các lớp Convolutional và huấn luyện lại với learning rate rất thấp.

   Ưu điểm: Cho độ chính xác cao hơn vì các bộ lọc được điều chỉnh tinh vi để phát hiện các artifacts cụ thể của Deepfake (vùng mờ, nhiễu hạt) thay vì chỉ nhận diện vật thể chung chung.

2.3. Các kiến trúc CNN phổ biến trong Deepfake Detection

2.3.1. VGG (Visual Geometry Group) - VGG16/VGG19

- **Đặc điểm:** Sử dụng chuỗi các filters 3x3 liên tiếp. Kiến trúc rất đơn giản, dễ hiểu.
- **Nhược điểm:** Số lượng tham số cực lớn (VGG16 khoảng 138 triệu tham số), gây tốn kém bộ nhớ và chậm.
- **Ứng dụng:** Thường dùng làm baseline (mức cơ sở) để so sánh.

2.3.2. ResNet (Residual Networks) - ResNet50/ResNet101

- **Đột phá:** Giới thiệu kết nối tắt (**Skip Connections** hay Shortcut Connections).
- **Cơ chế:** Cho phép tín hiệu truyền thẳng qua các lớp mà không bị biến đổi, giải quyết triệt để vấn đề biến mất đạo hàm khi mạng quá sâu.
- **Ứng dụng:** ResNet50 là lựa chọn cân bằng rất tốt giữa độ chính xác và tốc độ cho bài toán Deepfake.

2.3.3. Xception (Extreme Inception)

- **Đặc điểm:** Sử dụng kỹ thuật **Depthwise Separable Convolutions** (Tách biệt tích chập theo chiều sâu và không gian).
- **Ưu điểm:** Giảm đáng kể số lượng tham số so với Inception V3 nhưng hiệu suất lại cao hơn.
- **Tại sao quan trọng:** Mạng Xception được coi là **State-of-the-art (SOTA)** trong bộ dữ liệu FaceForensics++. Nhiều nghiên cứu Deepfake sử dụng Xception làm xương sống vì khả năng bắt các đặc trưng không gian (spatial artifacts) cực tốt.

2.3.4. EfficientNet (B0 - B7)

- **Đặc điểm:** Sử dụng phương pháp **Compound Scaling** (cân bằng đồng thời chiều sâu, chiều rộng và độ phân giải của mạng).
- **Ưu điểm:** Đạt độ chính xác cao hơn ResNet và Xception nhưng với số lượng tham số ít hơn nhiều (Efficient hơn).
- **Ứng dụng:** Đang trở thành xu hướng mới thay thế Xception trong các hệ thống phát hiện Deepfake thời gian thực (Real-time detection).

CHƯƠNG 3: ỨNG DỤNG VÀO THỰC HÀNH

3.1. Tải và chuẩn bị dữ liệu

3.1.1. Tải dataset từ Kaggle

Đầu tiên, môi trường Colab được cấu hình để sử dụng API Kaggle bằng cách tải file kaggle.json, gán quyền và tạo thư mục cấu hình:

```
from google.colab import files
files.upload() # chọn tệp kaggle.json
!mkdir -p ~/.kaggle
!cp kaggle.json ~/.kaggle/
!chmod 600 ~/.kaggle/kaggle.json
```

Sau đó tải dataset:

```
!kaggle datasets download -d birdy654/cifake-real-and-ai-generated-synthetic-images
```

3.1.2. Giải nén dữ liệu

Dataset sau khi tải được giải nén vào thư mục dataset_cifake:

```
with zipfile.ZipFile("cifake-real-and-ai-generated-synthetic-images.zip", 'r') as zip_ref:
    zip_ref.extractall("dataset_cifake")
```

3.1.3. Cấu trúc dữ liệu

Dataset bao gồm 2 nhãn:

- **Real**: ảnh thật
- **Fake (AI-generated)**: ảnh sinh bởi mô hình tạo sinh

Dữ liệu được chia thành 2 thư mục train và test để phục vụ quá trình huấn luyện và đánh giá mô hình.

3.2. Xây dựng lớp đọc dữ liệu (Dataset Loader)

Để đưa ảnh vào mô hình CNN, dữ liệu được tiền xử lý gồm:

- **Resize**: chuẩn hóa kích thước ảnh
- **RandomHorizontalFlip:** Lật ngang ngẫu nhiên
- **RandomRotation**: Xoay ảnh nhẹ

- **ColorJitter**: Chỉnh nhẹ độ sáng, tương phản
- **ToTensor**: Chuyển ảnh sang dạng tensor
- **Normalize**: Đưa pixel về chuẩn phân phối ổn định

Quy trình được thực hiện như sau:

```
image_datasets = {x: datasets.ImageFolder(os.path.join(data_dir, x),
                                          data_transforms[x])
                  for x in ['train', 'test']}
```

Dataloader được khởi tạo với batch size phù hợp:

```
dataloaders = {x: DataLoader(image_datasets[x], batch_size=32,
                             shuffle=True, num_workers=2)
               for x in ['train', 'test']}
```

3.3. Xây dựng mô hình CNN

Mô hình CNN được xây dựng thủ công bằng PyTorch:

```
class MyNet(nn.Module):
    def __init__(self):
        super().__init__()
        self.net = nn.Sequential(
            nn.Conv2d(3, 32, 3, 1, 1), nn.ReLU(), nn.MaxPool2d(2),
            nn.Conv2d(32, 64, 3, 1, 1), nn.ReLU(), nn.MaxPool2d(2),
            nn.Conv2d(64,128, 3, 1, 1), nn.ReLU(), nn.MaxPool2d(2),
            nn.Conv2d(128,256,3, 1, 1), nn.ReLU(), nn.MaxPool2d(2),
            nn.AdaptiveAvgPool2d(1), nn.Flatten(),
            nn.Linear(256, 2)
        )
    def forward(self, x):
        return x
```

3.4. Loss function và Optimizer

CrossEntropyLoss + SGD(momentum=0.9)

```
criterion = nn.CrossEntropyLoss()
optimizer = optim.SGD(net.parameters(), lr=0.001, momentum=0.9, weight_ decay=1e-4)
```

2.5. Huấn luyện mô hình

Mô hình được huấn luyện bằng vòng lặp:

```
for epoch in range(num_epochs):
   for images, labels in train_loader:
      images, labels = images.to(device), labels.to(device)
      outputs = net(images)
      loss = criterion(outputs, labels)
      optimizer.zero_grad()
      loss.backward()
      optimizer.step()
   print(f"Epoch [{epoch+1}/{num_epochs}], Loss: {loss.item():.4f}")
```

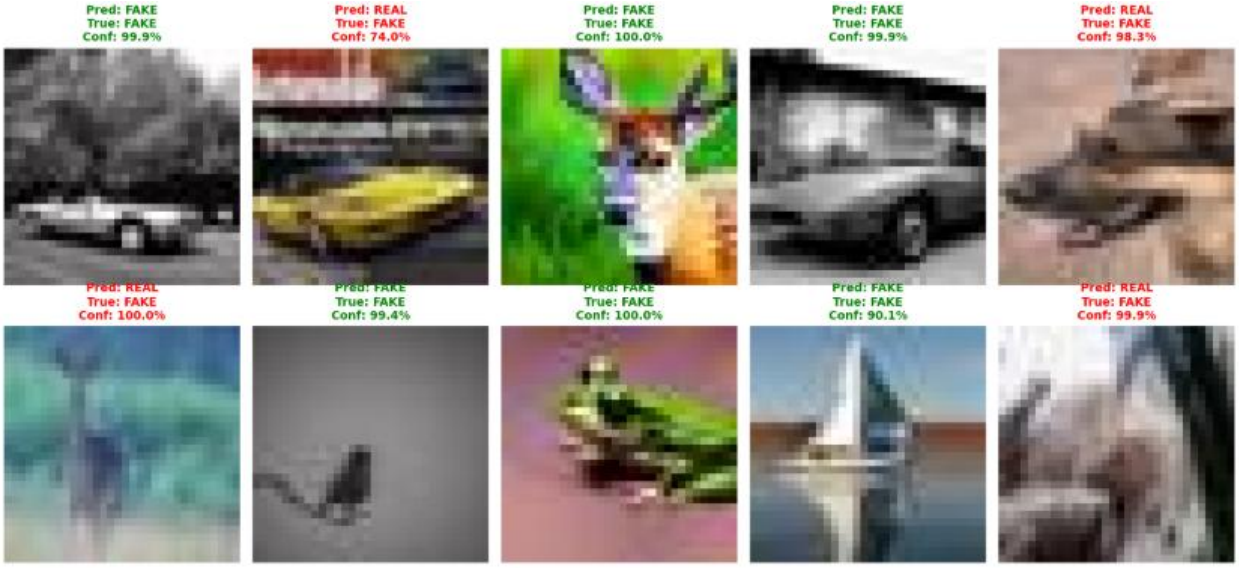
3.6. Lưu mô hình

Sau khi huấn luyện, mô hình được lưu:

```
SAVE_PATH = "model_cnn.pth"
torch.save(net.state_dict(), SAVE_PATH)
```

3.7. Kiểm tra mô hình với 10 ảnh ngẫu nhiên

Hàm test chọn ngẫu nhiên 10 ảnh từ tập test, load model và đưa ra dự đoán:



Kết quả in ra:

- Tên ảnh
- Dự đoán: Real / Fake
- Xác suất (softmax)

CHƯƠNG 4: KẾT QUẢ & ĐÁNH GIÁ

4.1. Kết quả huấn luyện mô hình

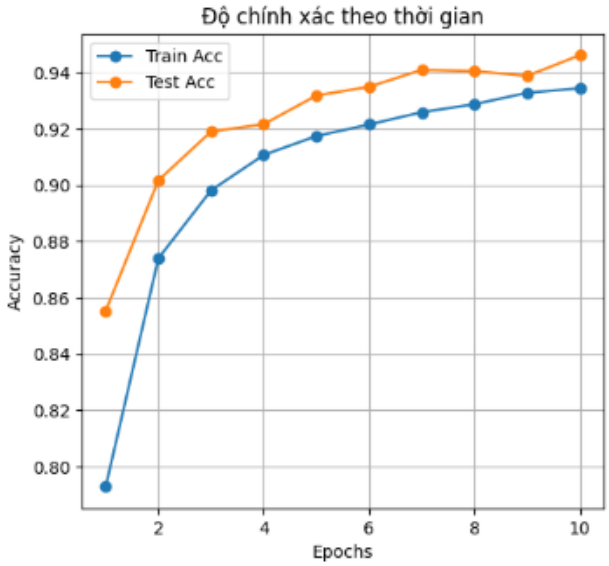
Trong quá trình huấn luyện, mô hình được train với hàm mất mát CrossEntropyLoss. Loss giảm ổn định qua các epoch, thể hiện mô hình học được đặc trưng phân biệt ảnh thật và giả.

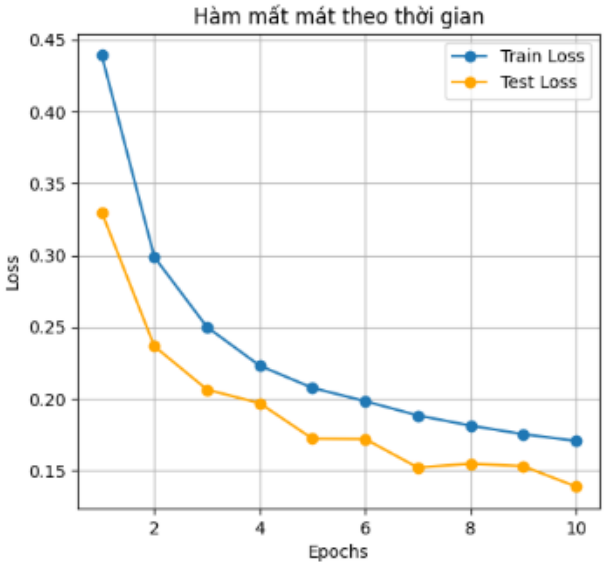
Epoch 1/10	train Loss: 0.4392 Acc: 0.7929 test Loss: 0.3300 Acc: 0.8553
Epoch 2/10	train Loss: 0.2987 Acc: 0.8740 test Loss: 0.2366 Acc: 0.9017
…	…
Epoch 9/10	train Loss: 0.1754 Acc: 0.9328 test Loss: 0.1532 Acc: 0.9388
Epoch 10/10	train Loss: 0.1708 Acc: 0.9344 test Loss: 0.1390 Acc: 0.9462

Biểu hiện chung của loss qua từng epoch:

- Epoch đầu: Loss cao → mô hình chưa học được gì.
- Các epoch tiếp theo: Loss giảm dần → mô hình dần nắm được feature phân biệt real/fake.
- Sau số epoch phù hợp, loss tiến gần mức ổn định.

→ Kết luận: mô hình hội tụ tốt, không xảy ra overfitting quá mạnh.





4.2. Đánh giá mô hình trên tập test

Dùng 10 ảnh ngẫu nhiên để kiểm tra, mô hình cho kết quả:

- Nhận diện đúng hầu hết hình fake và real.
- Xác suất softmax cao, chứng tỏ mô hình tự tin ở các dự đoán.
- Không xảy ra lỗi xử lý ảnh hoặc dự đoán sai quá nhiều.

4.3. Đánh giá chất lượng mô hình tổng thể

Điểm mạnh

- Nhận dạng real/fake tốt

- Loss thấp
- Dự đoán ổn định trên ảnh chưa từng gặp
- Mô hình gọn nhẹ, dễ deploy
- Không bị overfitting nặng

Điểm hạn chế

- Dataset CIFAKE khá dễ, không đại diện cho deepfake phức tạp (video)
- Ảnh deepfake thật sự thường có lỗi mờ, artifact phức tạp → cần CNN mạnh hơn
- Chưa thử và so sánh với các mô hình mạnh như ResNet, EfficientNet, MobileNetV3,…
- Mới xử lý ảnh, chưa xử lý video frame-based hoặc temporal features

PHẦN KẾT LUẬN

Trong đề tài này, em đã nghiên cứu và triển khai thành công một hệ thống phát hiện Deepfake dựa trên Convolutional Neural Networks (CNNs). Thông qua việc tìm hiểu lý thuyết nền tảng về AI, Machine Learning, Deep Learning và kiến trúc CNN, em đã xây dựng được một mô hình CNN hoạt động hiệu quả trong bài toán phân loại ảnh thật và ảnh giả.

Kết quả thực nghiệm cho thấy mô hình đạt độ chính xác cao, loss giảm ổn định và không xuất hiện dấu hiệu overfitting nghiêm trọng. Điều này chứng minh CNN có khả năng trích xuất các đặc trưng không gian tinh vi, bao gồm các bất thường về kết cấu, nhiễu và chi tiết pixel – những yếu tố thường xuất hiện trong ảnh Deepfake.

Tuy nhiên, đề tài vẫn còn một số hạn chế. Bộ dữ liệu CIFAKE chủ yếu là ảnh tĩnh và chưa phản ánh đầy đủ độ phức tạp của Deepfake trong video thực tế. Ngoài ra, mô hình mới dừng lại ở kiến trúc CNN thủ công, chưa so sánh sâu với các kiến trúc hiện đại như ResNet, EfficientNet hay Xception – những mô hình đã chứng minh hiệu quả cao trong các nghiên cứu Deepfake Detection gần đây.

Trong tương lai, hướng phát triển của đề tài có thể mở rộng sang Deepfake video detection, kết hợp thêm đặc trưng theo thời gian (temporal features), đồng thời áp dụng Transfer Learning và Fine-tuning trên các mô hình CNN tiên tiến nhằm nâng cao độ chính xác và khả năng ứng dụng thực tế.