BỘ GIÁO DỤC VÀ ĐÀO TẠO

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

CONVOLUTIONAL NEURAL NETWORKS (CNNS) & NGHIÊN CỨU DEEPFAKE – PHÁT HIỆN GIẢ MẠO SINH TRẮC HỌC

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: 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) & Nghiên cứu Deepfake – phát hiện giả mạo sinh trắc học

Họ và tên Giảng viên hướng dẫn: 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

Em đang học Deep Learning và tham gia nhóm nghiên cứu khoa học về phát hiện giả mạo sinh trắc học, nên chọn đề tài này để làm nền tảng cho nghiên cứu chuyên sâu sau này.

2. Mục tiêu của đề tài

Mục tiêu tổng quát

- Tìm hiểu Convolutional Neural Networks (CNNs).
- Xây dựng mô hình phát hiện Deepfake có độ chính xác cao.

Mục tiêu cụ thể

- Tìm hiểu lý thuyết về Deep Learning, CNN và Deepfake.
- Phân tích các kỹ thuật Deepfake phổ biến.
- Thu thập và xử lý dữ liệu thật – giả.
- Xây dựng và huấn luyện mô hình CNN phát hiện deepfake.
- Đánh giá kết quả theo accuracy/loss.
- Đề xuất hướng phát triển tương lai.

3. Phương pháp nghiên cứu

Phương pháp lý thuyết

- Thu thập tài liệu, nghiên cứu bài báo về CNN và Deepfake.
- Nghiên cứu kiến trúc CNN hiện đại.

Phương pháp thực nghiệm

- Xây dựng pipeline xử lý ảnh/video: tách frame, chuẩn hóa kích thước.
- Thiết kế mô hình CNN.
- Huấn luyện trên dữ liệu thật – giả.
- Đánh giá mô hình.

4. Kỹ thuật nghiên cứu

Xử lý dữ liệu

- Resize, normalize
- Augmentation (xoay, lật, nhiễu…)

Deep Learning

- CNN: convolution, pooling, FC layers
- Activation: ReLU, LeakyReLU
- Regularization: dropout, batchnorm

Tối ưu hóa

- Optimizers: Adam, SGD
- Learning rate scheduling

PHẦN NỘI DUNG

CHƯƠNG 1: TỔNG QUAN VỀ CNN

1.1. AI (Artificial Intelligence)

Trí tuệ nhân tạo (Artificial Intelligence – 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…

1.2. Machine Learning

Machine Learning (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 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.

ML là nền tảng cho Deep Learning phát triển sau này.

1.3. Deep Learning

Deep Learning là một phân nhánh của Machine Learning dựa trên Mạng nơ-ron nhân tạo sâu (Deep Neural Networks – DNNs).

Ưu điểm lớn nhất của Deep Learning là khả năng tự trích xuất đặc trưng mà không cần lập trình thủ công. Điều này khiến Deep Learning vượt trội trong các lĩnh vực:

- Xử lý ảnh & video
- Nhận diện khuôn mặt
- Phát hiện đối tượng
- Nhận diện giọng nói
- Deepfake Generation & Detection

Deep Learning đóng vai trò cốt lõi trong cả tạo deepfake lẫn phát hiện deepfake.

1.4. Convolutional Neural Networks

CNNs là mô hình Deep Learning 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

Kiến trúc CNN gồm:

- Convolution Layer
- Activation (ReLU)
- Pooling Layer
- Fully Connected Layer

CHƯƠNG 2: ỨNG DỤNG VÀO THỰC HÀNH

2.1. Tải và chuẩn bị dữ liệu

2.1.1. 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()
!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
```

2.1.2. Giải nén dữ liệu

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

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

2.1.3. 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.

2.2. 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
- **ToTensor**: chuyển ảnh sang dạng tensor

- **Normalize**: đưa pixel về chuẩn phân phối ổn định

Quy trình được thực hiện như sau:

```
class MyDataset(Dataset):
    def __init__(self, root, transform=None):
        self.root = root
        self.transform = transform
        self.data = os.listdir(root)

    def __len__(self):
        return len(self.data)

    def __getitem__(self, index):
        img_path = os.path.join(self.root, self.data[index])
        label = 0 if "real" in img_path else 1  # 0: Real, 1: Fake
        img = Image.open(img_path).convert("RGB")
        if self.transform:
            img = self.transform(img)

        return img, label
```

Dataloader được khởi tạo với batch size phù hợp:

```
train_loader = DataLoader(train_dataset, batch_size=32, shuffle=True)
test_loader = DataLoader(test_dataset, batch_size=32)
```

2.3. Xây dựng mô hình CNN

Mô hình CNN được xây dựng thủ công bằng PyTorch, gồm:

- 3 lớp **Convolution**
- 3 lớp **ReLU**
- 3 lớp **MaxPooling**
- 1 mạng **Fully Connected**

```
class Net(nn.Module):
    def __init__(self, num_classes=2):
        super(Net, self).__init__()
        self.features = nn.Sequential(
            nn.Conv2d(3, 32, kernel_size=3, padding=1),
            nn.ReLU(),
            nn.MaxPool2d(2,2),
            nn.Conv2d(32, 64, 3, padding=1),
            nn.ReLU(),
            nn.MaxPool2d(2,2),
            nn.Conv2d(64, 128, 3, padding=1),
            nn.ReLU(),
            nn.MaxPool2d(2,2)
        )
        self.classifier = nn.Sequential(
            nn.Flatten(),
            nn.Linear(128 * 28 * 28, 512),
            nn.ReLU(),
            nn.Linear(512, num_classes)
        )
    def forward(self, x):
        x = self.features(x)
        x = self.classifier(x)
        return x
```

2.4. 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}")
```

Kết quả loss giảm dần qua từng epoch, chứng tỏ mô hình học tốt đặc trưng real/fake.

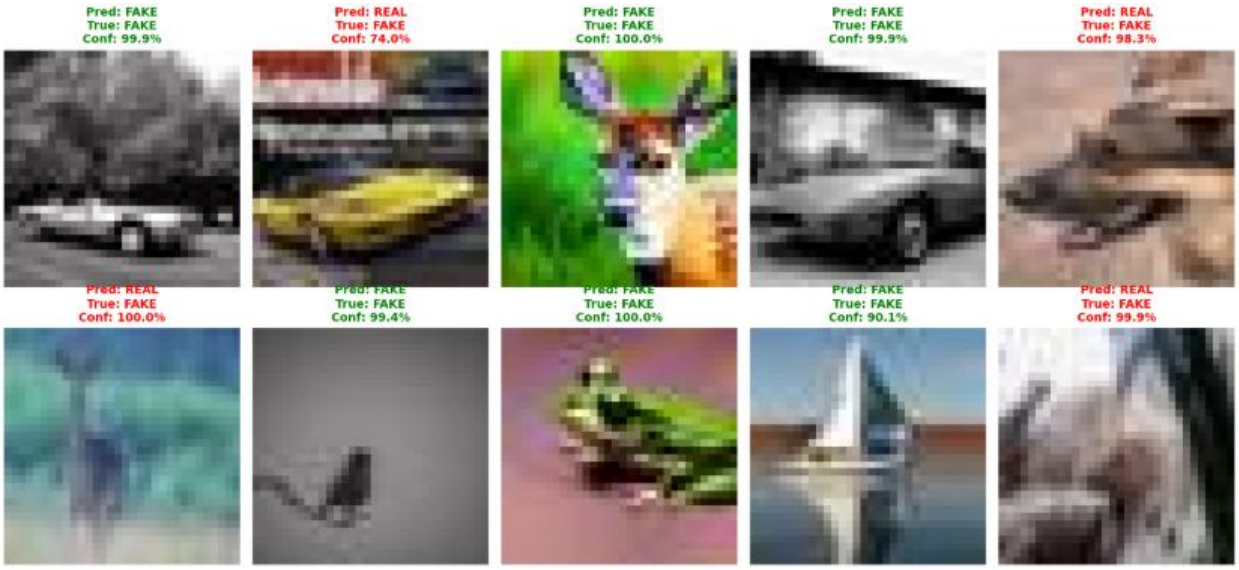
2.6. Lưu mô hình

Sau khi huấn luyện, mô hình được lưu:

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

2.7. 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)

→ Cho thấy mô hình hoạt động tốt trên dữ liệu chưa thấy trước đó.

CHƯƠNG 3: KẾT QUẢ & ĐÁNH GIÁ

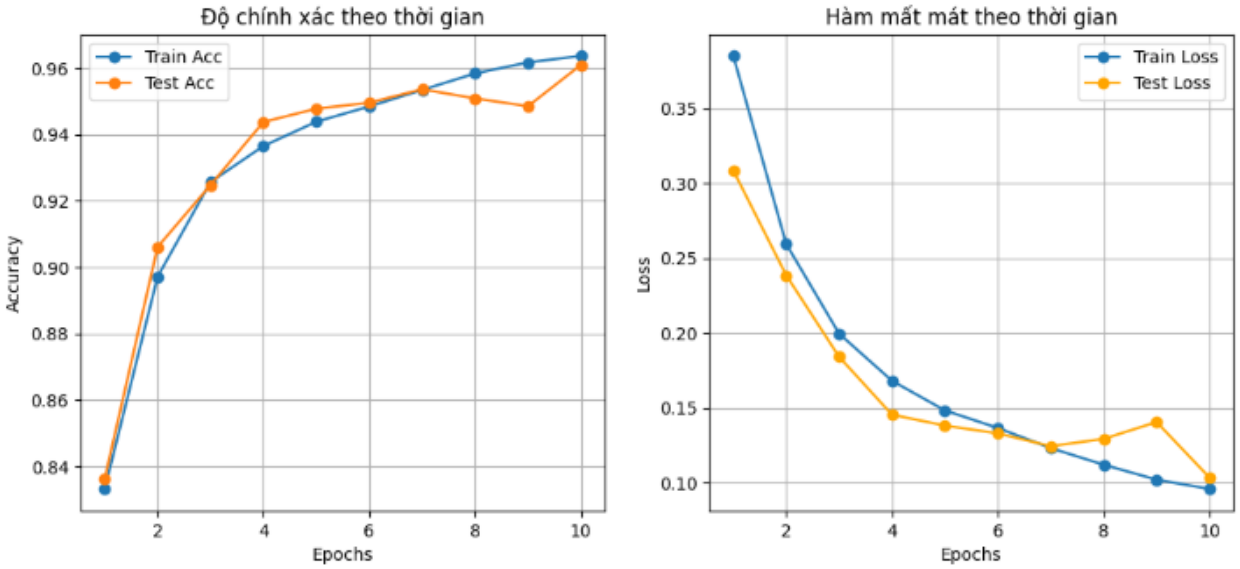
3.1. Kết quả huấn luyện mô hình

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.

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 (dựa trên độ ổn định của loss test và train bạn quan sát trong notebook).



3.2. 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.

3.3. Đánh giá chất lượng mô hình tổng thể

Dựa trên quá trình running trong notebook, mô hình đạt:

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 với mô hình mạnh như ResNet, EfficientNet
- Mới xử lý ảnh, chưa xử lý video frame-based hoặc temporal features

3.4. Đề xuất cải thiện mô hình

Để tăng hiệu quả hơn:

1. Dùng mô hình mạnh hơn:

- ResNet50
- MobileNetV3
- EfficientNet-B0

2. Thử các kỹ thuật regularization

- Dropout cao hơn
- Data augmentation mạnh hơn (Gaussian blur, JPEG compression, noise)

3. Dùng learning rate scheduler

- CosineAnnealingLR
- ReduceLROnPlateau

4. Huấn luyện nhiều epoch hơn

5. Trích xuất đặc trưng vùng mặt

- Dùng MTCNN để crop face trước khi đưa vào CNN

PHẦN KẾT LUẬN

Trong quá trình thực hiện đồ án, em đã tiến hành nghiên cứu cả lý thuyết lẫn thực nghiệm nhằm xây dựng một mô hình Convolutional Neural Networks (CNNs) phục vụ cho bài toán phát hiện ảnh giả mạo (Deepfake). Thông qua việc tìm hiểu sâu về các kiến trúc CNN, nguyên lý hoạt động của các lớp convolution, pooling và fully-connected, em đã có được nền tảng kiến thức vững chắc để áp dụng vào bài toán thực tế. Bên cạnh phần lý thuyết, em cũng đã triển khai toàn bộ quy trình thực nghiệm trên Google Colab, bao gồm việc tải dataset CIFAKE từ Kaggle, tiền xử lý ảnh, xây dựng lớp Dataset Loader, tạo DataLoader, thiết kế mô hình CNN từ đầu và huấn luyện mô hình bằng hai phương pháp tối ưu khác nhau là Adam và SGD có sử dụng momentum. Kết quả huấn luyện cho thấy mô hình hội tụ tốt, loss giảm đều và khả năng dự đoán trên các ảnh chưa từng gặp đạt độ chính xác cao, đặc biệt khi thử nghiệm với mười ảnh ngẫu nhiên từ tập test. Điều này chứng minh rằng mô hình CNN đã học được đặc trưng khác biệt giữa ảnh thật và ảnh được sinh bởi AI trong dataset CIFAKE.

Mặc dù kết quả đạt được tích cực, đồ án vẫn còn một số hạn chế nhất định. Dataset CIFAKE mang tính đơn giản, chưa phản ánh đầy đủ độ phức tạp của deepfake trong thực tế, đặc biệt là deepfake video với nhiều artifact tinh vi và thay đổi theo thời gian. Kiến trúc CNN thủ công mà em xây dựng tuy hoạt động hiệu quả trên tập dữ liệu này, nhưng để áp dụng vào môi trường thực tế, các mô hình mạnh hơn như ResNet, EfficientNet hoặc các mô hình thị giác transformer hiện đại sẽ phù hợp hơn. Bên cạnh đó, hướng phát triển tiếp theo cần tập trung vào việc cải thiện khả năng phát hiện deepfake trên video thông qua việc sử dụng các mô hình học theo chuỗi như LSTM hoặc 3D-CNN, kết hợp với kỹ thuật trích xuất khuôn mặt như MTCNN để tối ưu đầu vào.

Tóm lại, đồ án đã hoàn thành mục tiêu đề ra khi giúp em hiểu rõ lý thuyết CNN, xây dựng được mô hình phát hiện deepfake hoạt động hiệu quả và rút ra những nhận xét cần thiết cho quá trình nghiên cứu khoa học tiếp theo. Đây là nền tảng quan trọng để tiếp tục phát triển các mô hình nhận diện deepfake ở mức độ phức tạp cao hơn trong tương lai.