Hướng dẫn Huấn luyện Mô hình CNN với MNIST/CIFAR-10

1. Tải và Tiền Xử Lý Dữ liệu MNIST/CIFAR-10, Chia Thành Tập Train và Test Theo Tỷ Lệ 80:20

1.1. Tải Tập Dữ liệu MNIST/CIFAR-10

Giới thiệu về Datasets

- **MNIST**: Là tập dữ liệu gồm 70,000 hình ảnh chữ số viết tay (0-9), mỗi hình có kích thước 28x28 pixels, chia thành 60,000 ảnh cho huấn luyện và 10,000 ảnh cho kiểm tra.
- **CIFAR-10**: Là tập dữ liệu gồm 60,000 hình ảnh màu với 10 lớp khác nhau (như máy bay, ô tô, chim, mèo, v.v.), mỗi hình có kích thước 32x32 pixels, chia thành 50,000 ảnh cho huấn luyện và 10,000 ảnh cho kiểm tra.

Nguồn Tải:

- Thông qua Thư viện Học Máy: Các thư viện như TensorFlow, Keras, hoặc PyTorch thường cung cấp các hàm tích hợp để tải trực tiếp các tập dữ liệu này.
- Trang Web Chính Thức: Có thể tải dữ liệu từ các nguồn chính thức như Yann LeCun's website cho MNIST hoặc CIFAR-10 website cho CIFAR-10.

1.2. Tiền Xử Lý Dữ liệu

Đối với MNIST:

Chuyển Đổi Dữ Liệu Đầu Vào:

- Kích thước: Mỗi hình ảnh có kích thước 28x28 pixels. Cần chuyển đổi mỗi hình ảnh thành vector có kích thước 784 (28x28) nếu sử dụng mạng nơ-ron truyền thống hoặc giữ nguyên cấu trúc 2D nếu sử dụng CNN.

Chuẩn Hóa Giá Trị Pixel:

- Giá trị pixel ban đầu: Từ 0 đến 255
- Chuẩn hóa: Chia mỗi giá trị pixel cho 255 để đưa giá trị về khoảng [0,1]

Chuyển Labels Thành One-Hot Encoding:

- Khái niệm: Mỗi nhãn được biểu diễn bằng vector có kích thước bằng số lớp (10 lớp cho MNIST)
- Ví dụ: Nhãn "3" sẽ được biểu diễn thành [0, 0, 0, 1, 0, 0, 0, 0, 0, 0]

Đối với CIFAR-10:

Định Dạng Dữ Liệu:

- Kích thước: Mỗi hình ảnh có kích thước 32x32 pixels với 3 kênh màu (RGB)
- Chuyển đổi: Dữ liệu thường được giữ ở định dạng 3D (32x32x3) để khai thác được thông tin về màu sắc trong CNN

Chuẩn Hóa Giá Trị Pixel:

- Giá trị pixel ban đầu: Từ 0 đến 255
- Chuẩn hóa: Chia mỗi giá trị pixel cho 255 để đưa về khoảng [0,1]
- Tiếp tục chuẩn hóa (tuỳ chọn): Có thể trừ đi giá trị trung bình và chia cho độ lệch chuẩn của từng kênh màu

1.3. Chia Tập Train và Test Theo Tỷ Lệ 80:20

Đối với MNIST:

- Tổng số dữ liệu: 70,000 hình ảnh
- Chia tập:
    - Tập huấn luyện: 80% của dữ liệu (56,000 hình ảnh)
    - Tập kiểm tra: 20% của dữ liệu (14,000 hình ảnh)

Đối với CIFAR-10:

- Tổng số dữ liệu: 60,000 hình ảnh
- Chia tập:
    - Tập huấn luyện: 80% của dữ liệu (48,000 hình ảnh)
    - Tập kiểm tra: 20% của dữ liệu (12,000 hình ảnh)

Phương Pháp Chia Dữ liệu:

- Ngẫu Nhiên: Sử dụng phương pháp chia ngẫu nhiên để đảm bảo tính đa dạng
- Duy Trì Tỷ Lệ Lớp: Đảm bảo tỷ lệ các lớp trong cả hai tập là tương đương

2. Xây Dựng Mô Hình CNN

2.1. Thiết Kế Kiến Trúc Mạng CNN

1. Lớp Convolutional - 32 Filters:

- Chức năng: Trích xuất đặc trưng không gian từ dữ liệu đầu vào
- Thông số chính:
    - Số lượng filters: 32
    - Kích thước filter: 3x3 hoặc 5x5
    - Stride: Thường là 1
    - Padding: 'same' hoặc 'valid'

2. Lớp MaxPooling:

- Chức năng: Giảm kích thước không gian của đặc trưng
- Thông số chính:
    - Kích thước pool: 2x2

    - Stride: Thường bằng kích thước pool

3. Lớp Convolutional - 64 Filters:

- Chức năng: Trích xuất đặc trưng phức tạp hơn
- Thông số chính:
    - Số lượng filters: 64
    - Kích thước filter: 3x3 hoặc 5x5

4. Lớp MaxPooling:

- Kích thước pool: 2x2
- Stride: Bằng kích thước pool

5. Lớp Dense - 128 Neurons:

- Chức năng: Kết nối tất cả neuron để học đặc trưng phi tuyến tính
- Số lượng neuron: 128
- Hàm kích hoạt: ReLU

6. Lớp Output:

- Số lượng neuron: 10
- Hàm kích hoạt: Softmax

2.2. Khởi Tạo Trọng Số và Bias

Trọng số (Weights):

- Khởi tạo ngẫu nhiên với phân phối chuẩn
- Sử dụng phương pháp Xavier hoặc He Initialization

Bias:

- Khởi tạo bằng 0 hoặc giá trị nhỏ khác 0

2.3. Xác Định Hàm Mất Mát và Hàm Tối Ưu

Hàm Mất Mát:

- Cross-Entropy Loss cho phân loại đa lớp

Hàm Tối Ưu:

- Gradient Descent hoặc các biến thể
- Adam Optimizer được khuyến nghị

3. Huấn luyện và Đánh giá Mô hình

3.1. Huấn luyện Mô Hình

Quy Trình Huấn Luyện:

1. Forward Pass: Truyền dữ liệu qua mạng
2. Tính Hàm Mất Mát: So sánh dự đoán với nhãn thực tế
3. Backward Pass: Tính gradient và cập nhật trọng số
4. Lặp lại qua các epoch

3.2. Sử dụng Gradient Descent

Công thức Cập Nhật: $\theta = \theta - \eta \cdot \nabla \theta J(\theta)$

Trong đó:

- θ: trọng số và bias
- η: tốc độ học
- $\nabla \theta J(\theta)$: gradient của hàm mất mát

Lưu ý:

- Chọn tốc độ học phù hợp
- Cân nhắc sử dụng Mini-Batch Gradient Descent

3.3. Vẽ Đồ Thị và Đánh giá

Theo dõi và Ghi lại:

- Loss trên tập huấn luyện và kiểm tra
- Accuracy trên tập huấn luyện và kiểm tra

Vẽ đồ thị:

- Biểu diễn Loss và Accuracy theo epoch
- Sử dụng công cụ như Matplotlib

Tính Độ Chính Xác:

```
Accuracy = (Số lượng dự đoán đúng / Tổng số dự đoán) × 100%
```

Phân tích Kết quả:

- So sánh với các mô hình khác
- Xác định cần cải thiện những gì
- Đánh giá hiệu suất tổng thể