Hướng dẫn Cài đặt và So sánh Các Hàm Kích hoạt cho Mạng Nơ-ron

1. Cài đặt Các Hàm Kích Hoạt Cơ Bản và Đạo Hàm

1.1. Hiểu Rõ Các Hàm Kích Hoạt

Các hàm kích hoạt là các hàm phi tuyến được áp dụng trên đầu ra của mỗi neuron trong mạng nơ-ron. Chúng giúp mạng học được các mối quan hệ phức tạp giữa các đặc trưng dữ liệu.

Sigmoid (Logistic)

Công thức:

```
σ(x) = 1 / (1 + e^(-x))
```

Đạo hàm:

```
σ'(x) = σ(x) * (1 - σ(x))
```

Đặc điểm:

- Giá trị đầu ra nằm trong khoảng (0, 1)
- Thường được sử dụng trong các lớp đầu ra cho bài toán phân loại nhị phân

Tanh (Hyperbolic Tangent)

Công thức:

```
tanh(x) = (e^x - e^(-x)) / (e^x + e^(-x))
```

Đạo hàm:

```
tanh'(x) = 1 - tanh^2(x)
```

Đặc điểm:

- Giá trị đầu ra nằm trong khoảng (-1, 1)
- Thường được sử dụng trong các lớp ẩn
- Trung bình giá trị đầu ra là 0, giúp cải thiện hiệu suất huấn luyện

ReLU (Rectified Linear Unit)

Công thức:

```
ReLU(x) = max(0, x)
```

Đạo hàm:

```
ReLU'(x) = {
    1 nếu x > 0
    0 nếu x ≤ 0
}
```

Đặc điểm:

- Giá trị đầu ra không bị giới hạn ở phía trên
- Giảm thiểu vấn đề vanishing gradient
- Thường giúp mô hình hội tụ nhanh hơn

Leaky ReLU

Công thức:

```
Leaky ReLU(x) = {
    x     nếu x > 0
    αx    nếu x ≤ 0
}
```

(thường chọn α = 0.01)

Đạo hàm:

```
Leaky ReLU'(x) = {
    1    nếu x > 0
    α    nếu x ≤ 0
}
```

Đặc điểm:

- Giải quyết vấn đề "dying ReLU"
- Cho phép một phần gradient qua các giá trị âm
- Giữ cho các neuron không bị chết hoàn toàn trong quá trình huấn luyện

1.2. Cài Đặt Các Hàm Kích Hoạt và Đạo Hàm

1. Sigmoid và Đạo Hàm Sigmoid

- Mục tiêu: Tạo hàm nhận vào x và trả về σ(x) và σ'(x)
- Quy trình:
    - Tính giá trị sigmoid theo công thức
    - Tính đạo hàm từ giá trị sigmoid đã tính

2. Tanh và Đạo Hàm Tanh

- Mục tiêu: Tạo hàm nhận vào x và trả về tanh(x) và tanh'(x)
- Quy trình:
    - Tính giá trị tanh theo công thức
    - Tính đạo hàm từ giá trị tanh đã tính

3. ReLU và Đạo Hàm ReLU

- Mục tiêu: Tạo hàm nhận vào x và trả về max(0,x) và đạo hàm
- Quy trình:
    - Áp dụng hàm ReLU
    - Tính đạo hàm dựa trên giá trị x

4. Leaky ReLU và Đạo Hàm

- Mục tiêu: Tạo hàm nhận vào x và trả về Leaky ReLU(x) và đạo hàm
- Quy trình:
    - Áp dụng hàm Leaky ReLU với hệ số α
    - Tính đạo hàm dựa trên giá trị x

1.3. Vẽ Đồ Thị So Sánh

Mục Tiêu

Hiển thị đồ thị của các hàm kích hoạt và đạo hàm để hiểu rõ hành vi và đặc điểm.

Quy Trình

1. **Chọn Khoảng Giá Trị:**
    - Khoảng x từ -10 đến 10
    - Hiển thị hành vi ở cả giá trị âm và dương
2. **Tính Giá Trị:**
    - Tính giá trị hàm kích hoạt
    - Tính đạo hàm tương ứng
3. **Vẽ Đồ Thị:**
    - Sử dụng Matplotlib
    - Vẽ từng hàm và đạo hàm

- Sử dụng màu sắc khác nhau

4. **Phân Tích Đồ Thị:**
   - Sigmoid: Đầu ra (0,1), đạo hàm cực đại tại x=0
   - Tanh: Đầu ra (-1,1), đạo hàm tương tự sigmoid
   - ReLU: Đầu ra không giới hạn trên, đạo hàm đơn giản
   - Leaky ReLU: Tương tự ReLU nhưng có gradient cho x<0

2. Xây Dựng Mạng Nơ-ron Đơn Giản

2.1. Tạo Dữ Liệu Mẫu (2 Classes)

Quy Trình

1. **Chọn Loại Dữ Liệu:**
   - Không gian hai chiều
   - Hai lớp rõ ràng
2. **Tạo Tập Dữ Liệu:**
   - Lớp 1: Phân phối chuẩn quanh trung tâm 1
   - Lớp 2: Phân phối chuẩn quanh trung tâm 2
3. **Chia Tập Dữ Liệu:**
   - Training: 80%
   - Testing: 20%

2.2. Xây Dựng Mạng 2 Lớp

Cấu Trúc Chung

- **Lớp Input:** 2 neuron
- **Lớp Ẩn:** 4 neuron
- **Lớp Output:** 1 neuron (sigmoid)

Ba Phiên Bản

1. **Version 1 - Sigmoid:**
   - Hidden layer: Sigmoid
   - Output layer: Sigmoid
2. **Version 2 - ReLU:**
   - Hidden layer: ReLU
   - Output layer: Sigmoid
3. **Version 3 - Tanh:**

- Hidden layer: Tanh
- Output layer: Sigmoid

2.3. Khởi Tạo Tham Số

Trọng Số (Weights)

- Khởi tạo ngẫu nhiên với phân phối chuẩn
- Tránh giá trị quá lớn hoặc nhỏ

Bias

- Khởi tạo bằng 0 hoặc giá trị nhỏ
- Cho phép mô hình linh hoạt hơn

2.4. Hàm Mất Mát và Tối Ưu

Binary Cross-Entropy

```
L = -1/N Σ[yi*log(ŷi) + (1-yi)*log(1-ŷi)]
```

Trong đó:

- N: số mẫu
- yi: nhãn thực tế
- ŷi: giá trị dự đoán

Gradient Descent

- Learning rate (η): 0.01
- Cập nhật theo công thức: $\theta = \theta - \eta \nabla \theta J(\theta)$

3. So Sánh và Đánh Giá

3.1. Huấn Luyện

Quy Trình

1. Forward Pass
2. Tính Loss
3. Backward Pass
4. Cập nhật tham số
5. Lặp lại (100 epochs)

3.2. Đánh Giá

Vẽ Đồ Thị

- Loss function theo epoch
- So sánh 3 phiên bản

Tốc Độ Hội Tụ

- Số epoch để đạt ổn định
- So sánh tốc độ giữa các phiên bản

Độ Chính Xác

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

3.3. Báo Cáo Kết Quả

Cấu Trúc Báo Cáo

1. **Giới Thiệu**
    - Mục đích
    - Phương pháp
2. **Kết Quả**
    - Đồ thị Loss
    - Bảng độ chính xác
3. **Phân Tích**
    - So sánh hiệu suất
    - Ưu nhược điểm
4. **Kết Luận**
    - Tóm tắt phát hiện
    - Đề xuất cải tiến

Lưu Ý Chung

Thực Hành

- Hiểu rõ các khái niệm
- Kiểm tra dữ liệu kỹ
- Điều chỉnh siêu tham số
- Theo dõi quá trình huấn luyện
- Ghi chép kết quả chi tiết

Cải Tiến

- Thử nghiệm các cấu hình khác

- Tối ưu hóa siêu tham số
- Phân tích sâu kết quả