Deep learning

Mục Lục

1. Nền tảng và Khái niệm cơ bản

Deep Learning là một nhánh của Machine Learning (Học máy), trong đó các mô hình được thiết kế dựa trên mạng nơ-ron nhân tạo (Artificial Neural Networks - ANN). Điều làm Deep Learning khác biệt so với các kỹ thuật học máy khác là khả năng học từ dữ liệu thô và khám phá đặc trưng tự động, thay vì dựa vào việc thiết kế thủ công các đặc trưng (feature engineering).

Mạng Nơ-ron Nhân Tạo (ANN):

Mạng nơ-ron nhân tạo được lấy cảm hứng từ cách các tế bào thần kinh trong não hoạt động. Các mạng này gồm:

Neurons (nút):

Mỗi nút là một đơn vị tính toán.

Kết nối (Edges):

Mỗi kết nối giữa hai nút mang một trọng số (weight) thể hiện mức độ quan trọng.

Hàm kích hoạt (Activation Function):

Quyết định cách một nơ-ron phản ứng với dữ liệu đầu vào (ReLU, Sigmoid, Tanh, v.v.).

Học có giám sát (Supervised Learning):

Sử dụng dữ liệu có nhãn (labeled data) để huấn luyện mô hình.

Học không giám sát (Unsupervised Learning):

Dùng dữ liệu không có nhãn để khám phá cấu trúc (ví dụ: autoencoder).

Học tăng cường (Reinforcement Learning):

Học thông qua thử và sai, dựa trên phần thưởng hoặc phạt.

Dưới đây là 1 đoạn code cho 1 mạng nơron đơn giản.

```
import numpy as np
```

```
// Định nghĩa hàm kích hoạt sigmoid
def sigmoid(x):
  return 1 / (1 + np.exp(-x))

// Khởi tạo trọng số ngẫu nhiên
input_size = 2
hidden_size = 3
output_size = 1
weights1 = np.random.randn(input_size, hidden_size)
weights2 = np.random.randn(hidden_size, output_size)

// Hàm lan truyền tiến
def forward(X):
  hidden = sigmoid(np.dot(X, weights1))
  output = sigmoid(np.dot(hidden, weights2))
  return output

// Hàm huấn luyện (sử dụng gradient descent)
def train(X, y, epochs, learning_rate):
  for i in range(epochs):
    // Lan truyền tiến
    output = forward(X)

    // Tính lỗi
    error = y - output

    // Lan truyền ngược để tính gradient
    // ...

    // Cập nhật trọng số
    weights1 = # ...
    weights2 = # ...

// Dữ liệu huấn luyện (ví dụ)
X = np.array([, [11], [11], [11, 11]])
y = np.array([11, 11])

// Huấn luyện mạng
train(X, y, epochs=1000, learning_rate=0.1)

// Dự đoán
new_data = np.array([])
prediction = forward(new_data)
print(prediction)
```

2. Cấu trúc của Deep Neural Networks (DNNs)

a. Các tầng trong mạng:

Input Layer (Tầng đầu vào):

Nhận dữ liệu đầu vào (ví dụ: hình ảnh, văn bản, số liệu).

Hidden Layers (Tầng ẩn):

Xử lý và trích xuất đặc trưng thông qua các phép tính ma trận.

Output Layer (Tầng đầu ra):

Dự đoán kết quả cuối cùng (ví dụ: lớp nhãn cho phân loại).

b. Lan truyền tiến (Forward Propagation):

Dữ liệu di chuyển từ tầng đầu vào qua các tầng ẩn đến tầng đầu ra. Các phép tính bao gồm:

- Tích ma trận giữa đầu vào và trọng số.
- Cộng thêm giá trị bù (bias).
- Áp dụng hàm kích hoạt.

c. Lan truyền ngược (Backward Propagation):

Dựa trên lỗi giữa dự đoán và thực tế, các trọng số được cập nhật bằng cách lan truyền lỗi ngược qua mạng.

Gradient Descent:

Một thuật toán tối ưu để giảm lỗi qua từng lần huấn luyện (epoch).

Dưới đây là 1 đoạn code minh họa cấu trúc DNN đơn giản

```
from keras.models import Sequential
from keras.layers import Dense

// Khởi tạo mô hình DNN
model = Sequential()

// Thêm tầng đầu vào và tầng ẩn thứ nhất
model.add(Dense(units=64, activation='relu', input_dim=10))

// Thêm tầng ẩn thứ hai
model.add(Dense(units=32, activation='relu'))

// Thêm tầng đầu ra
model.add(Dense(units=1, activation='sigmoid'))

// In ra tóm tắt kiến trúc mạng
model.summary()
```

```
Giải thích đoạn code:
- from keras.models import Sequential và from keras.layers import Dense: Import
các lớp cần thiết từ thư viện Keras.
- model = Sequential(): Khởi tạo một mô hình DNN theo kiểu Sequential, nghĩa là
các tầng được xếp chồng lên nhau theo thứ tự.
```

```
- model.add(Dense(units=64, activation='relu', input_dim=10)): Thêm tầng đầu tiên
với:
o units=64: 64 nơ-ron.
o activation='relu': Hàm kích hoạt ReLU.
o input_dim=10: Số lượng đặc trưng đầu vào là 10.
- model.add(Dense(units=32, activation='relu')): Thêm tầng ẩn thứ hai tương tự như
tầng ẩn thứ nhất nhưng với 32 nơ-ron.
- model.add(Dense(units=1, activation='sigmoid')): Thêm tầng đầu ra với:
o units=1: 1 nơ-ron đầu ra (cho bài toán phân loại nhị phân).
o activation='sigmoid': Hàm kích hoạt sigmoid.
- model.summary(): In ra tóm tắt kiến trúc mạng DNN.
```

3. Các kiến trúc mạng phổ biến

a. Mạng tích chập (Convolutional Neural Networks - CNNs):

Ứng dụng:

Xử lý hình ảnh (nhận diện khuôn mặt, xe tự lái, phân loại ảnh y tế).

Hoạt động:

Áp dụng bộ lọc (filters) để trích xuất đặc trưng từ ảnh.

Pooling Layer:

Giảm kích thước ảnh nhưng giữ lại thông tin quan trọng.

Flattening:

Chuyển từ dạng ma trận thành vector để đưa vào các tầng dày đặc (Dense Layers).

b. Mạng hồi quy (Recurrent Neural Networks - RNNs):

Ứng dụng:

Xử lý dữ liệu tuần tự (văn bản, chuỗi thời gian, âm thanh).

Hoạt động:

RNN có khả năng nhớ trạng thái trước đó thông qua các vòng lặp bên trong.

Biến thể:

LSTM (Long Short-Term Memory) và GRU (Gated Recurrent Units) khắc phục vấn đề "quên" ở RNN truyền thống.

c. Transformers:

Ứng dụng:

Xử lý ngôn ngữ tự nhiên (NLP) và học sâu tổng quát (ChatGPT, BERT).

Cơ chế Attention:

Tập trung vào các phần quan trọng nhất trong chuỗi dữ liệu.

d. Generative Adversarial Networks (GANs):

Ứng dụng:

Tạo hình ảnh, video, hoặc âm thanh giả trông như thật.

Cấu trúc: Gồm hai mạng:

- Generator: Tạo ra dữ liệu giả.
- Discriminator: Phân biệt dữ liệu giả và thật.

4. Quy trình Huấn luyện một mô hình Deep Learning

Chuẩn bị dữ liệu:

- Thu thập dữ liệu từ thực tế hoặc bộ dữ liệu sẵn có (MNIST, CIFAR-10, ImageNet).
- Tiền xử lý (xử lý nhiễu, chuẩn hóa, tăng cường dữ liệu). Xây dựng mô hình:
- Chọn kiến trúc phù hợp với bài toán. Định nghĩa các thông số (số lượng tầng, số nơ-ron mỗi tầng, loại hàm kích hoạt).
- Chia dữ liệu thành tập huấn luyện, tập kiểm tra, và tập xác minh.
- Đo hiệu suất bằng các chỉ số như độ chính xác, độ lỗi, F1-score.
- Tích hợp mô hình vào các ứng dụng thực tế.

5. Ứng dụng thực tiễn

Y tế:

Phát hiện bệnh qua hình ảnh (X-quang, MRI), dự đoán kết quả điều trị.

Xe tự lái:

Xử lý hình ảnh và ra quyết định trong thời gian thực.

Thương mại:

Cá nhân hóa trải nghiệm người dùng, hệ thống gợi ý (Netflix, Amazon).

An ninh:

Nhận diện khuôn mặt, giám sát qua camera.

Thanks for watching!