# Xây dựng và huấn luyện một mô hình mạng nơ-ron nhân tạo đơn giản để phân loại dữ liệu ảnh từ bộ dữ liệu MNIST

Các thuật toán và các bước được thực hiện:

1. Tải dữ liệu MNIST:

* MNIST là một bộ dữ liệu nổi tiếng trong học sâu, chứa 60,000 ảnh huấn luyện và 10,000 ảnh kiểm tra. Mỗi ảnh là một số viết tay, kích thước 28x28 pixel.
* X\_train và X\_test chứa ảnh, trong khi y\_train và y\_test chứa nhãn tương ứng.

1. Chuẩn hóa dữ liệu và chuyển đổi nhãn thành dạng one-hot:

* Chuẩn hóa ảnh: Các giá trị pixel của ảnh nằm trong khoảng [0, 255]. Để việc huấn luyện hiệu quả hơn, ta chuẩn hóa các giá trị pixel này về khoảng [0, 1] bằng cách chia cho 255.0.
* Chuyển nhãn thành dạng one-hot: Dữ liệu nhãn (y\_train và y\_test) ban đầu là các số nguyên từ 0 đến 9. Ta sử dụng to\_categorical để chuyển chúng thành dạng one-hot encoding, tức là mỗi nhãn sẽ được biểu diễn bằng một vector có 10 phần tử, trong đó một phần tử là 1 và các phần tử còn lại là 0.

1. Chia tập dữ liệu thành tập huấn luyện và tập validation:

* Tạo tập validation: Từ tập huấn luyện ban đầu, ta chia ra một phần (50,000 ảnh) để làm tập huấn luyện và một phần còn lại (10,000 ảnh) để làm tập validation.
* Tập validation sẽ được sử dụng để đánh giá mô hình trong quá trình huấn luyện và điều chỉnh tham số.

1. Xây dựng mô hình mạng nơ-ron:

* Mô hình Sequential: Đây là một mô hình tuần tự, tức là các lớp trong mô hình được nối tiếp nhau.
  1. Flatten: Chuyển ảnh đầu vào 2D (28x28) thành vector 1D có 784 phần tử (28 \* 28).
  2. Dense(1024, activation='relu'): Lớp ẩn có 1024 nút, sử dụng hàm kích hoạt ReLU (Rectified Linear Unit). ReLU là hàm kích hoạt phổ biến trong các mô hình học sâu vì nó giúp giảm bớt vấn đề vanishing gradient.
  3. Dense(10, activation='softmax'): Lớp đầu ra có 10 nút (tương ứng với 10 lớp từ 0 đến 9), sử dụng hàm kích hoạt Softmax để xuất ra xác suất cho từng lớp. Softmax sẽ biến đổi giá trị đầu ra của mạng nơ-ron thành một phân phối xác suất.

1. Biên dịch mô hình:

* Optimizer: Sử dụng SGD (Stochastic Gradient Descent) với tốc độ học learning\_rate=0.008. SGD là thuật toán tối ưu hóa truyền thống và hiệu quả trong học máy.
* Loss function: Sử dụng categorical\_crossentropy vì đây là bài toán phân loại đa lớp (10 lớp trong trường hợp này).
* Metrics: Mô hình sẽ theo dõi accuracy (độ chính xác) trong quá trình huấn luyện và đánh giá.

1. Huấn luyện mô hình:

* Epochs: Huấn luyện mô hình trong 10 vòng lặp (epochs). Mỗi vòng lặp sử dụng toàn bộ tập huấn luyện để cập nhật trọng số của mô hình.
* Batch size: Chia tập huấn luyện thành các lô nhỏ (batch size = 256), giúp giảm bộ nhớ và tăng tốc quá trình huấn luyện.
* Validation: Tập dữ liệu validation được sử dụng để theo dõi độ chính xác của mô hình trong mỗi epoch.

1. Đánh giá mô hình trên tập test:

* Sau khi huấn luyện, mô hình được đánh giá trên tập test (chưa tham gia vào quá trình huấn luyện), giúp kiểm tra độ chính xác tổng quát của mô hình trên dữ liệu mới.

1. Trực quan hóa trọng số của lớp ẩn đầu tiên:

* Trọng số của lớp ẩn đầu tiên (lớp Dense(1024, activation='relu')) được truy xuất và trực quan hóa. Các trọng số này đại diện cho các kết nối giữa các nút trong lớp đầu vào (784 phần tử) và lớp ẩn (1024 nút).
* Trực quan hóa trọng số giúp hiểu cách mà mô hình học được từ dữ liệu đầu vào.

1. Vẽ biểu đồ Accuracy và Loss trong quá trình huấn luyện:

* Accuracy: Biểu đồ mô tả sự thay đổi của độ chính xác trong quá trình huấn luyện và trên tập validation.
* Loss: Biểu đồ mô tả sự thay đổi của hàm mất mát trong quá trình huấn luyện và trên tập validation.