SO SÁNH HIỆU NĂNG CỦA THUẬT TOÁN BACKPROPAGATION VÀ FORWARD-FORWARD TRONG BÀI TOÁN NHẬN DẠNG ẢNH

COMPARE THE PERFORMANCE OF BACKPROPAGATION AND FORWARD-FORWARD ALGORITHM IN IMAGE RECOGNITION PROBLEM

Trần Thanh Hùng[1,*], Đỗ Ngọc Sơn[1]

DOI: http://doi.org/10.57001/huih5804.2025.062

TÓM TẮT

Trong bài báo này, chúng tôi sẽ so sánh hiệu năng của hai thuật toán học sâu phổ biến là Backpropagation và Forward-Forward trong bài toán nhận dạng ảnh sử dụng tập dữ liệu MNIST. Chúng tôi sẽ đánh giá độ chính xác và tốc độ huấn luyện của cả hai thuật toán. Kết quả cho thấy, Backpropagation đạt độ chính xác cao hơn trong khi Forward-Forward có thể có tốc độ huấn luyện nhanh hơn trong một số trường hợp.

Từ khoá: *Backpropagation; mạng nơ-ron; Forward-Forward; Forward; Backward.*

ABSTRACT

In this article, we will compare the performance of two algorithms, Backpropagation and Forward-Forward, in image recognition problems using MNIST data. We will evaluate the accuracy and training speed of both algorithms. The results show that Backpropagation achieves higher accuracy when Forward-Forward can have faster training speed in some cases.

Keywords: *Backpropagation; Neural Network; Forward-Forward; Forward; Backward.*

[1]Trường Đại học Công nghiệp Hà Nội
[*]Email: hungtt@fit-haui.edu.vn
Ngày nhận bài: 04/11/2024
Ngày nhận bài sửa sau phản biện: 15/3/2025
Ngày chấp nhận đăng: 28/3/2025

1. GIỚI THIỆU

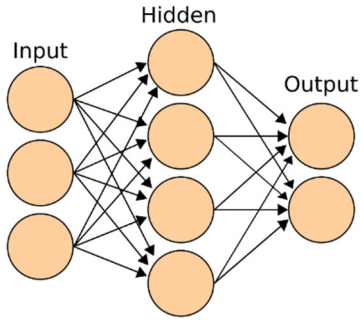
Nhận dạng ảnh là một trong những ứng dụng quan trọng của học sâu, đặc biệt là trong các bài toán phân loại ảnh. Thuật toán Backpropagation đã chứng minh được hiệu quả cao trong việc huấn luyện các mạng nơ-ron sâu. Tuy nhiên, do quá trình tính toán gradient phức tạp, Backpropagation có thể tốn nhiều tài nguyên tính toán. Thuật toán Forward-Forward, mặc dù còn khá mới và chưa được nghiên cứu sâu, được đề xuất như một phương pháp tiềm năng để giảm bớt gánh nặng tính toán.

2. THUẬT TOÁN BACKPROPAGATION

2.1. Giới thiệu về thuật toán Backpropagation

Thuật toán Backpropagation[1], hay còn gọi là thuật toán lan truyền ngược, là một phương pháp học có giám sát chủ yếu được sử dụng để huấn luyện mạng nơ-ron nhân tạo. Thuật toán này được phát triển vào những năm 1970 và trở nên phổ biến vào những năm 1980 sau khi được mô tả chi tiết bởi Rumelhart, Hinton và Williams trong công trình "Learning representations by back-propagating errors".

Backpropagation là một thuật toán tối ưu hóa được sử dụng rộng rãi trong việc huấn luyện các mạng nơ-ron. Thuật toán này dựa trên việc tính toán gradient của hàm mất mát với các trọng số của mạng nơ-ron và điều chỉnh các trọng số này thông qua quy tắc gradient descent.



Hình 1. Mô hình mạng nơ-ron nhân tạo

2.2. Nguyên lý hoạt động

Backpropagation hoạt động dựa trên nguyên lý lan truyền lỗi ngược từ đầu ra về đầu vào của mạng nơ-ron. Quá trình này bao gồm hai giai đoạn chính:

1. Lan truyền tiến (Forward Propagation):

○ Đầu vào được truyền qua các lớp của mạng nơ-ron để tính toán đầu ra dự đoán.

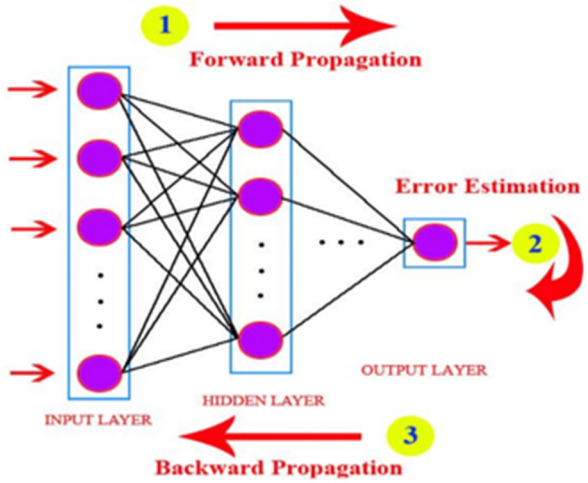
○ Các trọng số của mạng giữ nguyên trong giai đoạn này.

2. Lan truyền ngược (Backward Propagation):

○ Tính toán lỗi giữa đầu ra dự đoán và giá trị đầu ra thực tế bằng một hàm mất mát (loss function).

○ Sử dụng đạo hàm của hàm mất mát để tính toán gradient của lỗi đối với mỗi trọng số của mạng.

○ Cập nhật các trọng số theo hướng giảm gradient để giảm thiểu lỗi.



Hình 2. Hình ảnh minh hoạ lan truyền tiến và lan truyền ngược trong thuật toán Backpropagation

2.3. Thuật toán cụ thể

Backpropagation sử dụng quy tắc Gradient Descent để cập nhật các trọng số của mạng nơ-ron. Quy trình chi tiết như sau:

- Giả sử mạng nơ-ron có cấu trúc như sau:

• **Lớp đầu vào: x**

• **Lớp ẩn:** Neurons **h** với hàm kích hoạt **σ**

• **Lớp đầu ra:** Neurons **y** với hàm kích hoạt **σ**

- Lan truyền tiến (Forward Propagation):

• **Tính toán đầu ra của mạng:** Đầu tiên, tính toán giá trị đầu ra của mô hình cho mỗi mẫu dữ liệu đầu vào bằng cách đi qua từng lớp của mạng nơ-ron.

• **Tính toán hàm mất mát:** So sánh giá trị dự đoán với giá trị thực tế để tính toán hàm mất mát.

○ Tính toán giá trị lớp ẩn:

$$h = \sigma(W.x + b) \quad (1)$$

○ Tính toán giá trị lớp đầu ra:

$$y_{pred} = \sigma(V.h + b_y) \quad (2)$$

- Tính Gradient của Hàm mất mát:

• **Tính đạo hàm của hàm mất mát theo đầu ra:** Đo lường sự thay đổi của hàm mất mát với sự thay đổi của đầu ra dự đoán.

• **Tính đạo hàm theo trọng số:** Sử dụng quy tắc chuỗi để tính gradient của hàm mất mát theo trọng số của mỗi lớp.

○ Tính gradient của hàm mất mát theo đầu ra:

$$\frac{\partial L}{\partial y_{pred}} = y_{pred} - y_{true} \quad (3)$$

○ Tính gradient của hàm mất mát theo trọng số lớp đầu ra:

$$\frac{\partial L}{\partial V} = \frac{\partial L}{\partial y_{pred}} . \frac{\partial y_{pred}}{\partial V} \quad (4)$$

○ Tính gradient của hàm mất mát theo trọng số lớp ẩn:

$$\frac{\partial L}{\partial W} = \frac{\partial L}{\partial h} . \frac{\partial h}{\partial W} \quad (5)$$

- Lan truyền ngược (Backward Propagation):

• **Tính gradient cho từng lớp:** Tính toán gradient của hàm mất mát theo từng trọng số và bias của mỗi lớp từ lớp đầu ra trở lại lớp đầu vào.

• **Cập nhật trọng số:** Dựa trên gradient tính được, cập nhật trọng số và bias của mô hình bằng cách sử dụng một phương pháp tối ưu hóa, chẳng hạn như Gradient Descent.

○ Cập nhật trọng số lớp đầu ra:

$$V \leftarrow V - \mu . \frac{\delta L}{\delta V} \quad (6)$$

○ Cập nhật trọng số lớp ẩn:

$$W \leftarrow W - \mu . \frac{\delta L}{\delta W} \quad (7)$$

Với μ là tốc độ học của mạng (Learning rate).

- Lặp lại quá trình:

• **Lặp lại quá trình:** Lặp lại các bước trên cho nhiều lần (epochs) để mô hình ngày càng cải thiện hiệu suất của nó.

Backpropagation giúp cải thiện mô hình bằng cách sử dụng gradient để điều chỉnh trọng số nhằm giảm thiểu lỗi. Quá trình này dựa vào việc tính toán đạo hàm của hàm mất mát theo trọng số và cập nhật chúng một cách liên tục trong quá trình huấn luyện.

2.4. Xây dựng cấu trúc mô hình thuật toán

Chúng tôi sử dụng một mạng nơ-ron với cấu trúc gồm các lớp convolutional và fully connected như sau:

• Lớp Conv2D với 32 filter, kernel size 3x3 và hàm kích hoạt ReLU

• Lớp MaxPooling2D với pool size 2x2

• Lớp Conv2D với 64 filter, kernel size 3x3 và hàm kích hoạt ReLU

• Lớp MaxPooling2D với pool size 2x2

• Lớp Conv2D với 64 filter, kernel size 3x3 và hàm kích hoạt ReLU

• Lớp Flatten

• Lớp Dense với 64 đơn vị và hàm kích hoạt ReLU

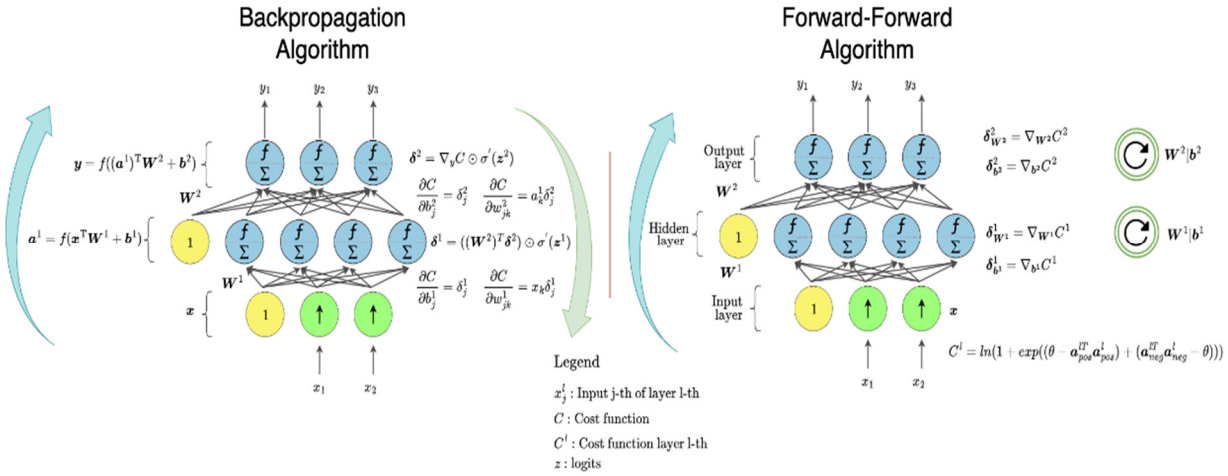
• Lớp Dense với 10 đơn vị và hàm kích hoạt softmax

3. THUẬT TOÁN FORWARD-FORWARD

3.1. Giới thiệu về thuật toán Forward-Forward

Thuật toán "Forward-Forward" (FF) [3] là một phương pháp mới được đề xuất bởi Geoffrey Hinton, một trong những nhà tiên phong trong lĩnh vực học sâu (deep learning) và mạng nơ-ron. Phương pháp này là một phần trong các nỗ lực của ông nhằm tìm ra các phương pháp thay thế cho thuật toán Backpropagation truyền thống, nhằm giải quyết một số hạn chế và thách thức của nó.

Forward-Forward Algorithm là một cách tiếp cận mới để huấn luyện mạng nơ-ron mà không sử dụng Backpropagation. Thay vì lan truyền ngược lỗi để cập nhật trọng số, thuật toán này thực hiện hai pass lan truyền tiến: một pass với dữ liệu thực (positive pass) và một pass với dữ liệu giả hoặc nhiễu (negative pass). Mục tiêu của phương pháp này là làm cho các kích hoạt của các lớp ẩn với dữ liệu thực trở nên lớn hơn so với dữ liệu giả.



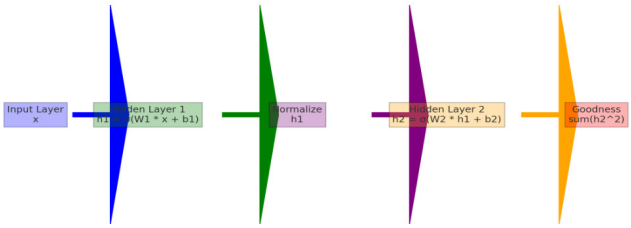
Hình 3. Hình ảnh minh hoạ nguyên lý hoạt động của 2 thuật toán Backpropagation Algorithm và Forward-Forward algorithm

3.2. Nguyên lý hoạt động của thuật toán FF

- Positive Pass**:**

Positive Pass là bước trong thuật toán Forward-Forward mà dữ liệu thực được đưa vào mạng và các kích hoạt của các lớp ẩn được tính toán. Mục tiêu của bước này là làm cho độ tốt của dữ liệu thực cao hơn so với dữ liệu giả.

Giả sử chúng ta có hai lớp ẩn và sử dụng tổng của các hoạt động bình phương làm thước đo độ tốt. Trong bước Positive Pass, dữ liệu thực **x** được đưa vào mạng.



Hình 4. Minh hoạ quá trình Possive Pass trong một mạng nơ-ron với 2 lớp ẩn

Các bước của **Positive Pass:**

$$x \rightarrow \text{Mạng Neural} \rightarrow h_1 \rightarrow \tilde{h}_1 \rightarrow h_2 \rightarrow ..$$

○ Bước 1. Kích hoạt của lớp thứ nhất

$$h_1 = \sigma(W_1.x + b_1) \quad (8)$$

○ Bước 2. Chuẩn hoá vector hoạt động của lớp thứ nhất

$$\tilde{h}_1 = \frac{h_1}{\|h_1\|} \quad (9)$$

○ Bước 3. Kích hoạt của lớp thứ hai

$$h_2 = \sigma(W_2\tilde{h}_1 + b_2) \quad (10)$$

○ Bước 4. Đo lường độ tốt của lớp thứ hai

$$\text{Goodness} = \sum_i h_{2,i}^2 \quad (11)$$

○ Bước 5. Cập nhật trọng số: Trọng số của mạng được điều chỉnh sao cho độ tốt của dữ liệu thực tăng lên (Positive (real) data).

$$W \leftarrow W + \mu . \nabla_w \text{Goodness} \quad (12)$$

Trong đó, μ là tốc độ học (Learning rate) và $\nabla_w \text{Goodness}$ là gradient của độ tốt đối với trọng số W.

Positive Pass giúp đảm bảo rằng mạng nơ-ron học cách tối đa hóa độ tốt với dữ liệu thực, làm cho mạng nơ-ron trở nên hiệu quả hơn trong việc phân loại và nhận diện các mẫu dữ liệu chính xác.

- Negative Pass

Negative Pass là bước trong thuật toán Forward-Forward mà dữ liệu giả hoặc nhiễu được đưa vào mạng và các kích hoạt của các lớp ẩn được tính toán. Mục tiêu của bước này là làm cho độ tốt của dữ liệu giả nhỏ hơn độ tốt của dữ liệu thực.

Giả sử chúng ta có hai lớp ẩn và sử dụng tổng của các hoạt động bình phương làm thước đo độ tốt. Trong bước Negative Pass, dữ liệu giả $\tilde{x}$ được đưa vào mạng.

Các bước của **Negative Pass:**

$\tilde{x} \rightarrow \text{Mạng Neural} \rightarrow h_1 \rightarrow \tilde{h}_1 \rightarrow h_2 \rightarrow \ldots$

• Bước 1. Kích hoạt của lớp thứ nhất

$$h_1 = \sigma(W_1.\tilde{x} + b_1) \quad (13)$$

• Bước 2. Chuẩn hoá vector hoạt động của lớp thứ nhất

$$\tilde{h}_1 = \frac{h_1}{\|h_1\|} \quad (14)$$

• Bước 3. Kích hoạt của lớp thứ hai

$$h_2 = \sigma(W_2\tilde{h}_1 + b_2) \quad (15)$$

• Bước 4. Đo lường độ tốt của lớp thứ hai

$$\text{Goodness} = \sum_i h_{2,i}^2 \quad (16)$$

• Bước 5. Cập nhật trọng số: Trọng số của mạng được điều chỉnh sao cho độ tốt của dữ liệu giả giảm xuống dưới một ngưỡng nhất định.

$$W \leftarrow W - \mu.\nabla_w \text{Goodness} \quad (17)$$

Negative Pass giúp đảm bảo rằng mạng nơ-ron không chỉ học cách tối đa hóa độ tốt với dữ liệu thực mà còn học cách giảm thiểu độ tốt với dữ liệu giả. Điều này làm cho mạng nơ-ron trở nên mạnh mẽ hơn và ít bị lừa bởi các mẫu dữ liệu không phù hợp.

3.3. Xây dựng cấu trúc mô hình thuật toán FF

Chúng tôi sử dụng một mạng nơ-ron đơn giản với cấu trúc Fully connected như sau:

• Lớp đầu vào với 784 đơn vị (28x28 pixel)

• Lớp ẩn với 128 đơn vị và hàm kích hoạt ReLU

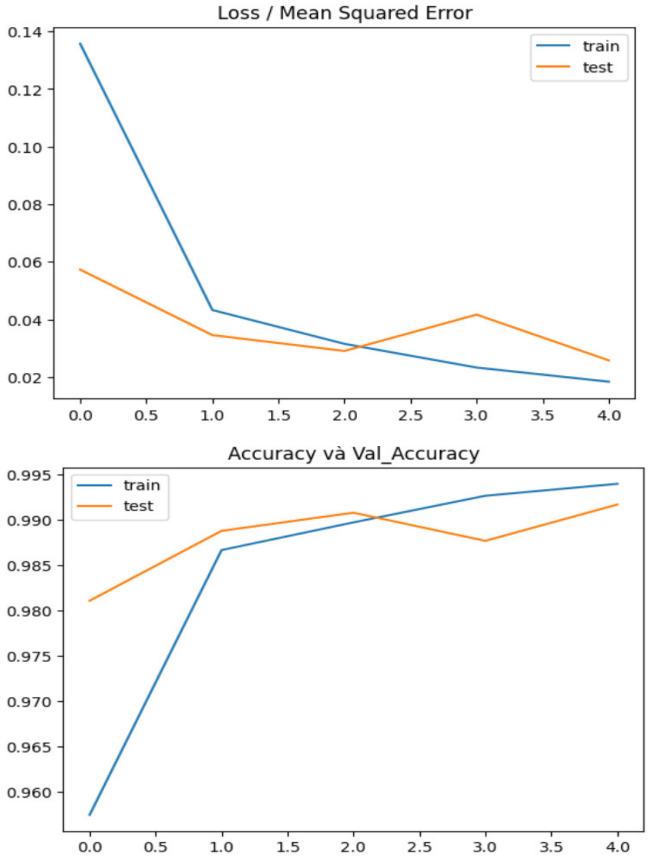
• Lớp đầu ra với 10 đơn vị và hàm kích hoạt softmax.

4. KẾT QUẢ NGHIÊN CỨU

Trong bài toán nhận dạng ảnh sử dụng tập dữ liệu MNIST, được chia 60000 ảnh cho tập dữ liệu train, và 10000 ảnh cho tập dữ liệu test.

Với giải thuật Backpropagation, thời gian để huấn luyện bộ dữ liệu với số epochs = 5 là: 323,4694390296936 giây.

Độ chính xác của giải thuật qua các biểu đồ giá trị của hàm mất mát (loss/val_loss) và biểu đồ val_loss và accuracy/val_accuracy như thể hiện trong hình 5.



Hình 5. Biểu đồ hàm loss/val_loss, accuracy/val_accuracy của thuật toán Backpropagation

Với giải thuật Forward-Forward, thời gian để huấn luyện bộ dữ liệu với số epochs = 100 là: 134,54301190376282 giây.

Hình 6. Biểu đồ hàm loss/val_loss/val_accuracy của thuật toán Forward-Forward

Độ chính xác của giải thuật qua các biểu đồ giá trị của hàm mất mát (loss/val_loss) và biểu đồ val_loss, và accuracy/val_accuracy như thể hiện trong hình 6.

5. KẾT LUẬN

- Độ chính xác của 2 giải thuật

• **Backpropagation:** Thuật toán Backpropagation đạt độ chính xác rất cao, thường trên 98% trên tập dữ liệu kiểm tra MNIST, độ chính xác của thuật toán Forward-Forward còn tương đối thấp, đạt khoảng 79%. Điều này cho thấy khả năng mạnh mẽ của Backpropagation trong việc tối ưu hóa và học từ dữ liệu.

• **Forward-Forward:** Độ chính xác của thuật toán Forward-Forward thấp hơn so với Backpropagation. Điều này là do cách cập nhật trọng số đơn giản hơn và thiếu khả năng tối ưu hóa mạnh mẽ.

- Tốc độ huấn luyện của 2 giải thuật

• **Backpropagation:** Mặc dù đạt được độ chính xác cao, Backpropagation tốn nhiều thời gian hơn do phải tính toán gradient và truyền ngược qua mỗi lớp. Trong thử nghiệm của chúng tôi, thời gian huấn luyện là khoảng vài phút.

• **Forward-Forward:** Thuật toán Forward-Forward có thể huấn luyện nhanh hơn do bỏ qua quá trình tính toán gradient phức tạp.

TÀI LIỆU THAM KHẢO

[1]. Rumelhart D. E., Hinton G. E., Williams R. J., "Learning representations by back-propagating errors," *Nature*, 323(6088), 533-536, 1986

[2]. Goodfellow I., Bengio Y., Courville A., *Deep Learning*. MIT Press, 2016.

[3]. Hinton G., *The Forward-Forward Algorithm: Theoretical Foundations and Practical Applications*. 2022. arXiv:2212.13345

[4]. Sun T., Li J., "Applications of the forward-forward algorithm in image recognition," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR 2023)*, 987-995, 2023.

[5]. Zhang Y., Wang X., "Exploring the efficiency of the forward-forward algorithm in deep neural networks," *Neural Networks,* 156, 45-60, 2023.

[6]. Nguyen M., Chen L., "A review of the forward-forward algorithm: Challenges and future directions," *Artificial Intelligence Review*, 74, 789-810, 2023.

AUTHORS INFORMATION

Tran Thanh Hung, Do Ngoc Son

Hanoi University of Industry, Vietnam