



THUẬT TOÁN CNN TRONG NHẬN DIỆN KÝ TỰ BIỂN SỐ XE

THÀNH VIÊN
NGUYỄN HỮU LỘC - B22DCCN508
NGUYỄN VĂN HỌC - B22DCCN352

TIỀN XỬ LÝ

1

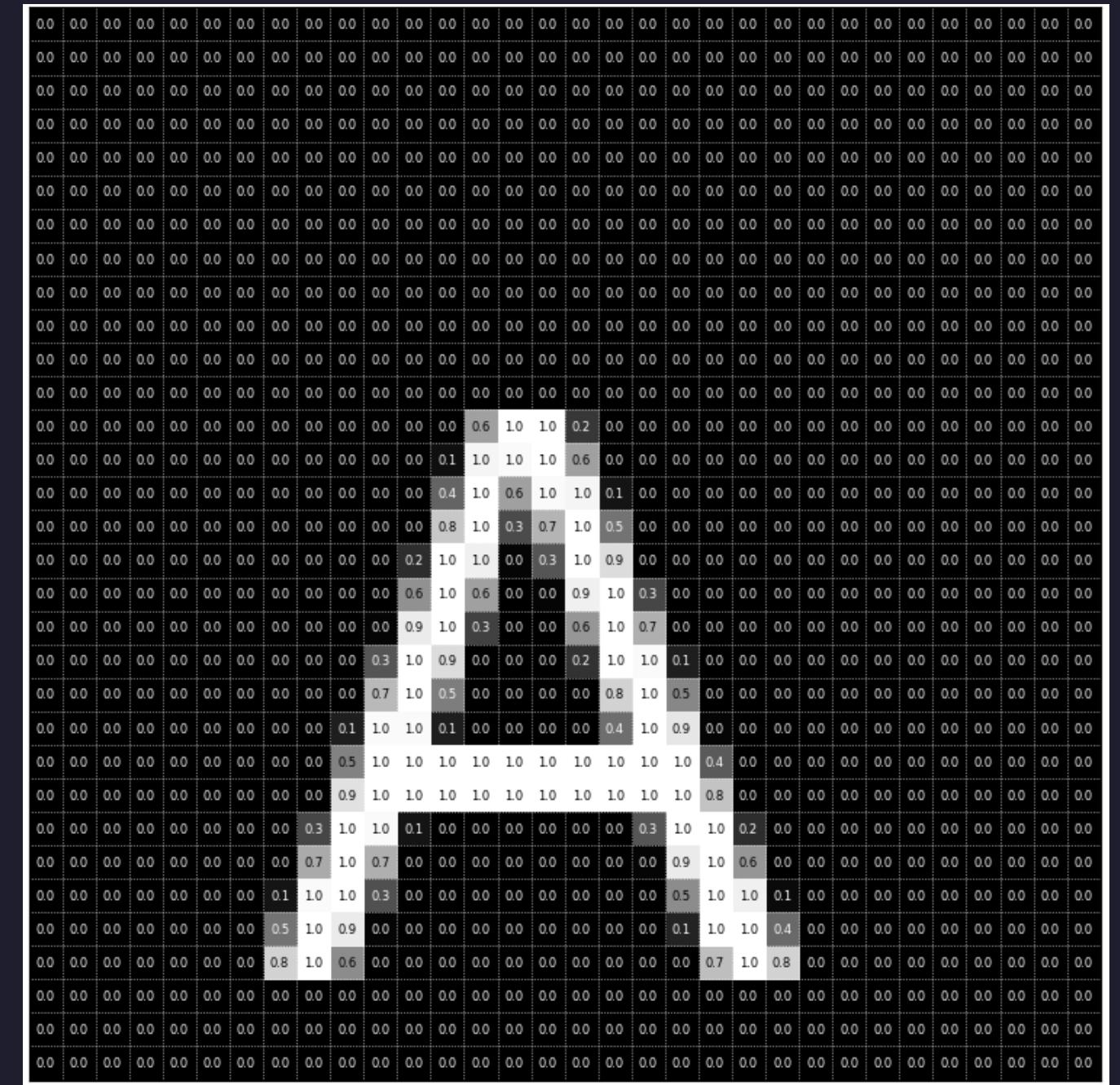
Chuyển ảnh gốc thành ảnh xám

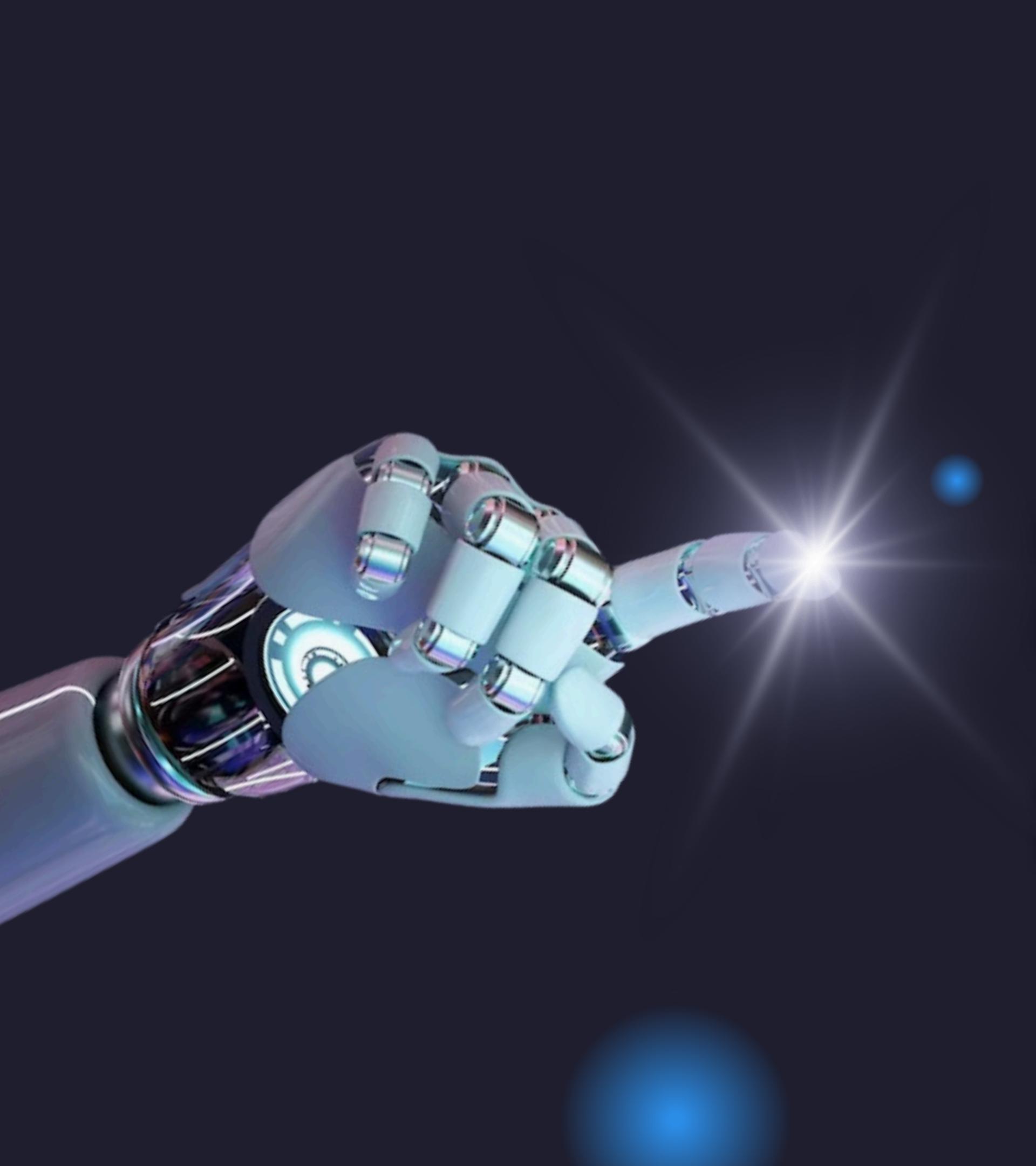
2

Chuẩn hoá kích thước về 28x28

3

Chuẩn hoá giá trị mỗi pixel nằm trong đoạn [0,1]





FORWARD PROPAGATION

QUÁ TRÌNH MÁY TÍNH
HỌC TẬP

LAYER CONVOLUTIONAL

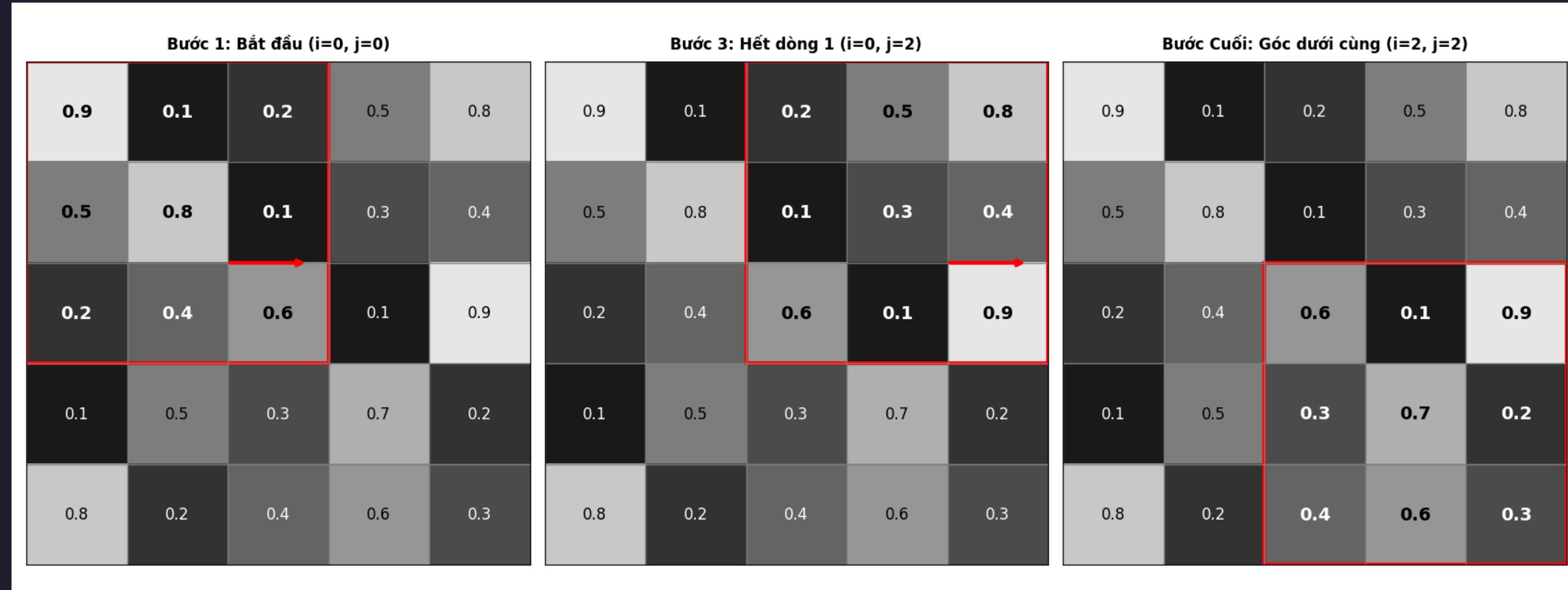
Sử dụng 8 bộ lọc (filters) để quét và trích xuất các đặc trưng cơ bản của ảnh như đường nét, góc cạnh.

input: 1 ảnh 28x28.

output: (26, 26, 8)

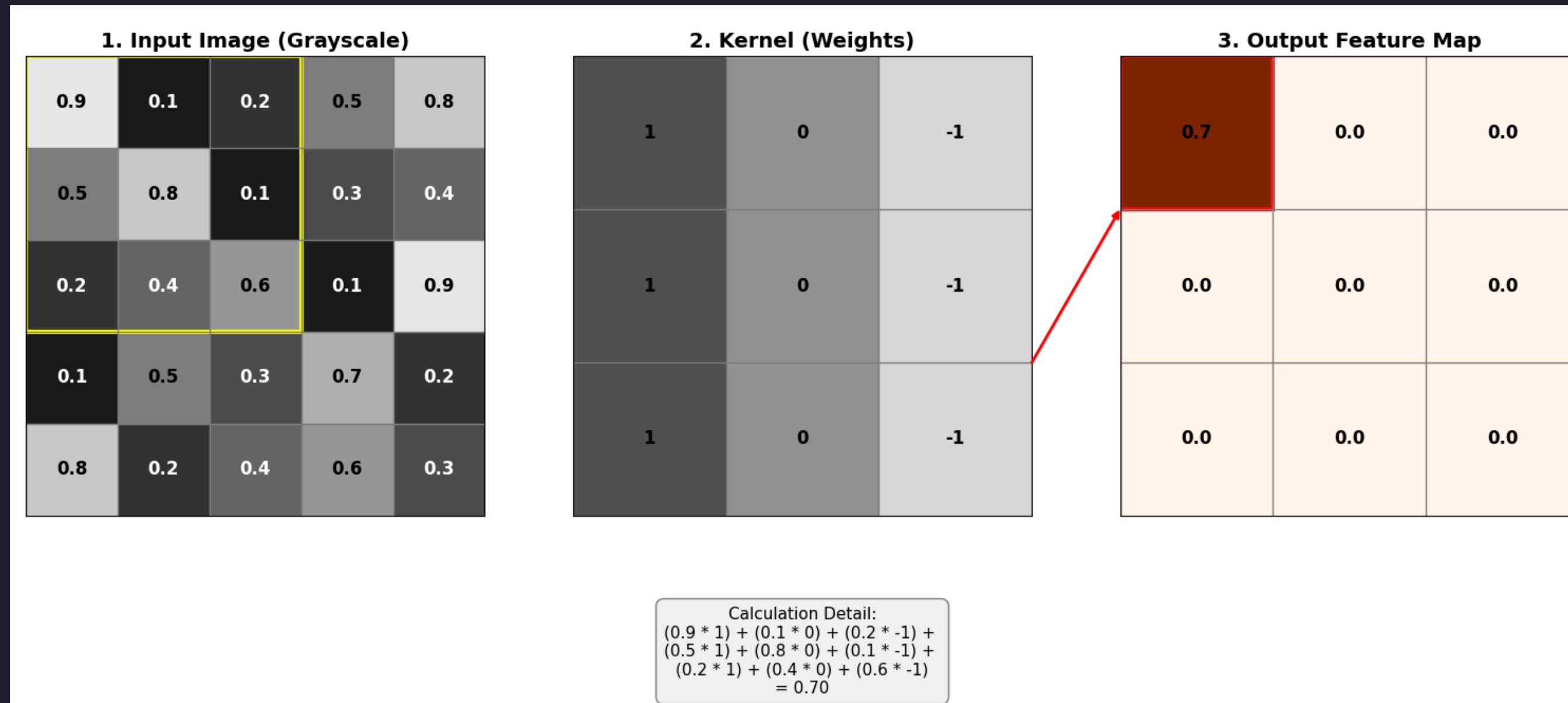
LAYER CONVOLUTIONAL

Nhận đầu vào là 1 ảnh 28x28. Duyệt qua từng vị trí có thể đặt kernel



LAYER CONVOLUTIONAL

Tính tích chập cho từng vùng trên ảnh gốc để đạt được ảnh đầu ra



LAYER RELU

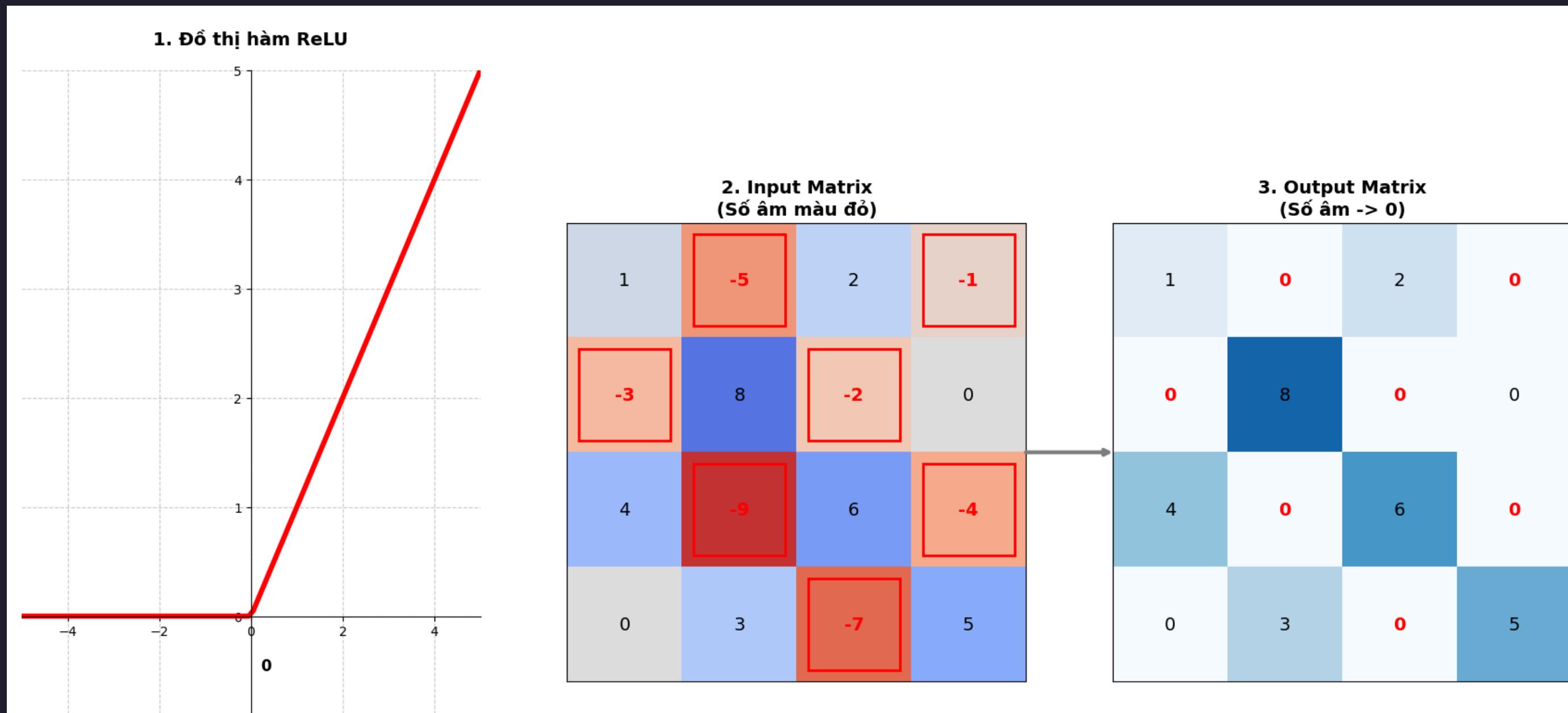
Loại bỏ các tín hiệu yếu hoặc nhiễu (các vùng màu đen hoặc không khớp với bộ lọc), chỉ giữ lại những nơi thực sự tìm thấy nét chữ.

input: (26, 26, 8)

output: (26, 26, 8)

LAYER RELU

Duyệt qua từng giá trị của khối dữ liệu đầu vào và áp dụng hàm ReLU



LAYER MAX POOLING

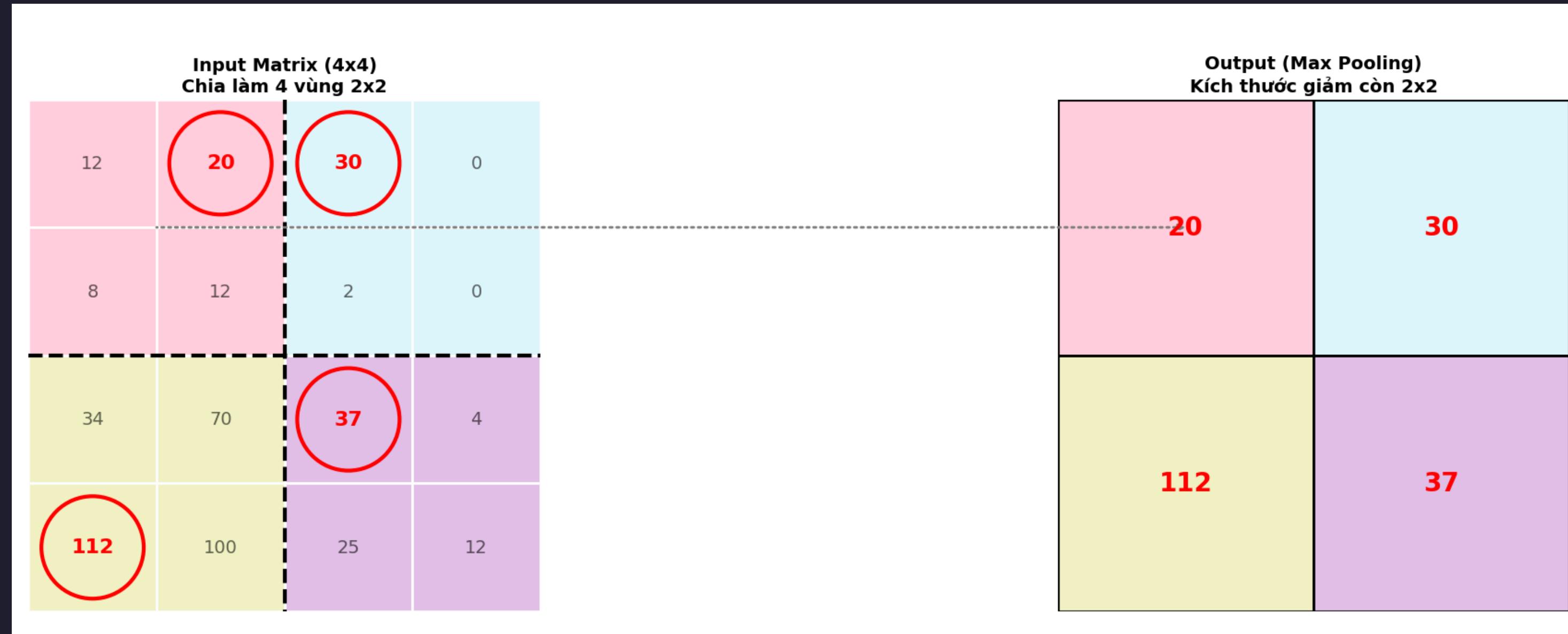
Giảm lượng dữ liệu cần tính toán. Giữ lại đặc trưng nổi bật nhất

input: (26, 26, 8)

output: (13, 13, 8)

LAYER MAX POOLING

Dùng cửa sổ 2×2 duyệt qua dữ liệu, trong mỗi cửa sổ chỉ chọn ra 1 giá trị lớn nhất



LAYER FLATTEN

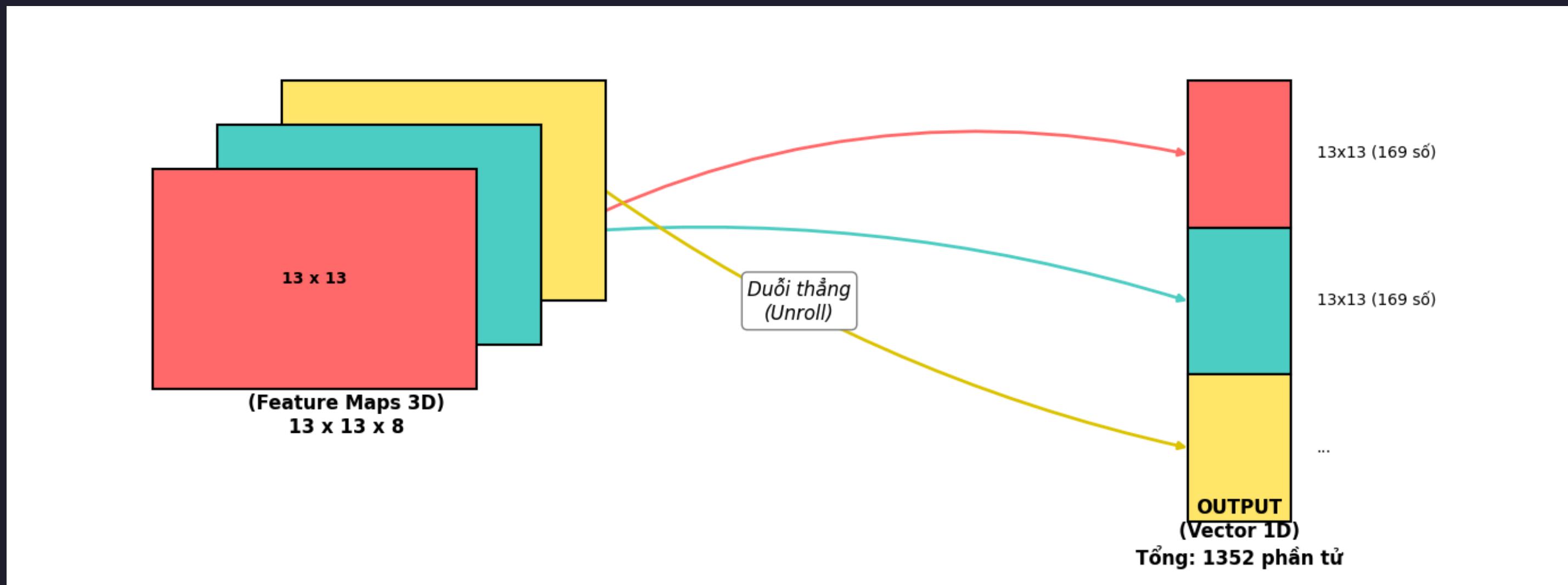
Duỗi phẳng dữ liệu từ không gian 3 chiều thành vector 1 chiều, làm đầu vào cho lớp Kết nối
đầy đủ (Fully Connected) phía sau

input: (13, 13, 8)

output: vector (1352)

LAYER FLATTEN

Gỡ các lớp của khối 3D và xếp chúng thành một hàng dọc duy nhất.



LAYER DENSE 1

Tại đây, các đặc điểm rời rạc (nét cong góc trái, nét thẳng ở giữa...) được tổ hợp lại để hình thành các khái niệm phức tạp hơn (ví dụ: "có một vòng tròn ở trên và nét móc ở dưới").

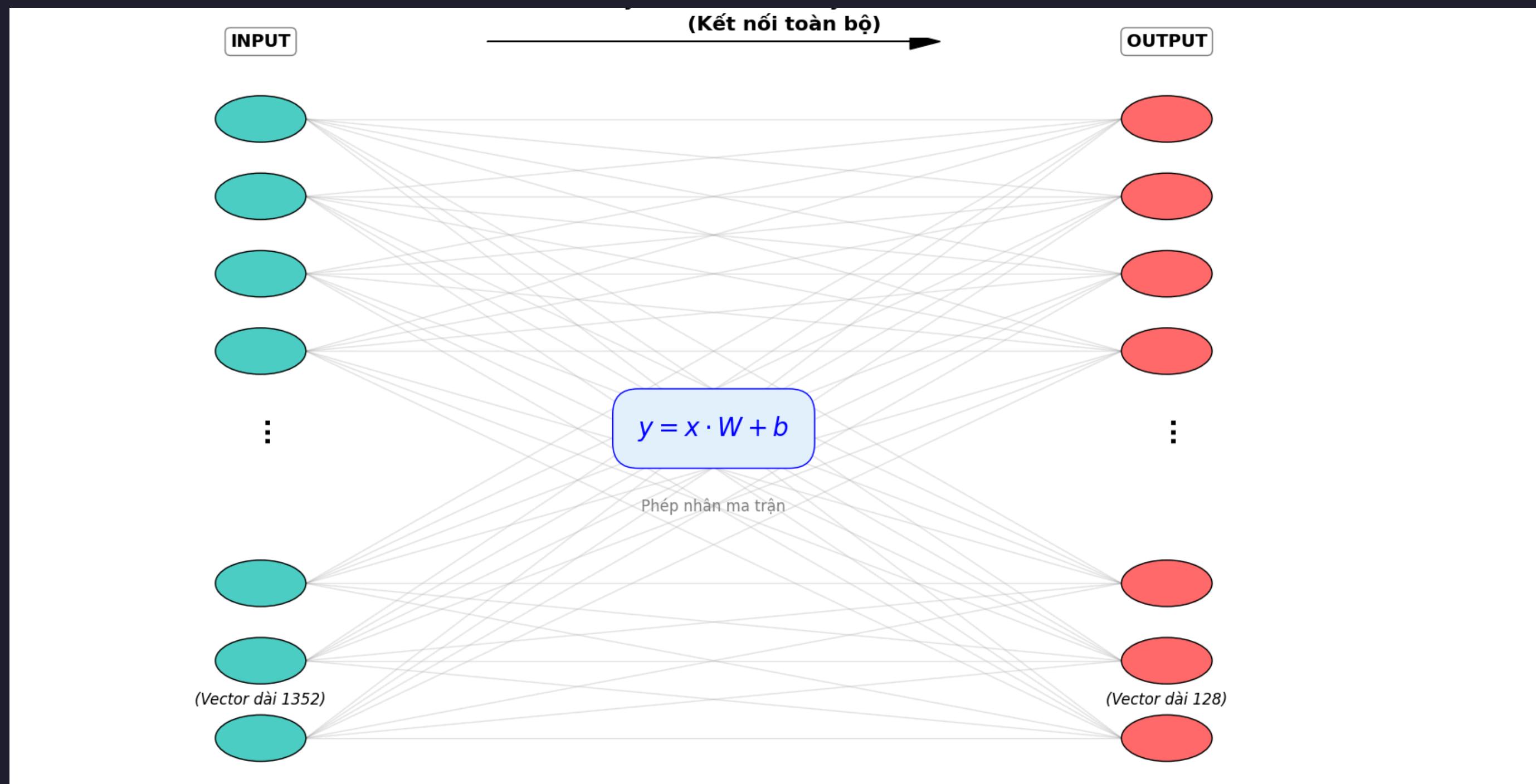
input: vector (1352)

output: vector (128)

LAYER DENSE 1

Thực hiện phép nhân ma trận: $y = x \cdot W + b$.

Kết nối 1352 điểm dữ liệu đầu vào với 128 nơ-ron ẩn (hidden nodes).



LAYER RELU & DROPOUT

Dropout giống như việc bắt mô hình "học bịt mắt". Nó buộc mạng không được phụ thuộc quá nhiều vào bất kỳ một manh mối duy nhất nào, giúp mô hình thông minh hơn và tránh học vẹt (Overfitting).

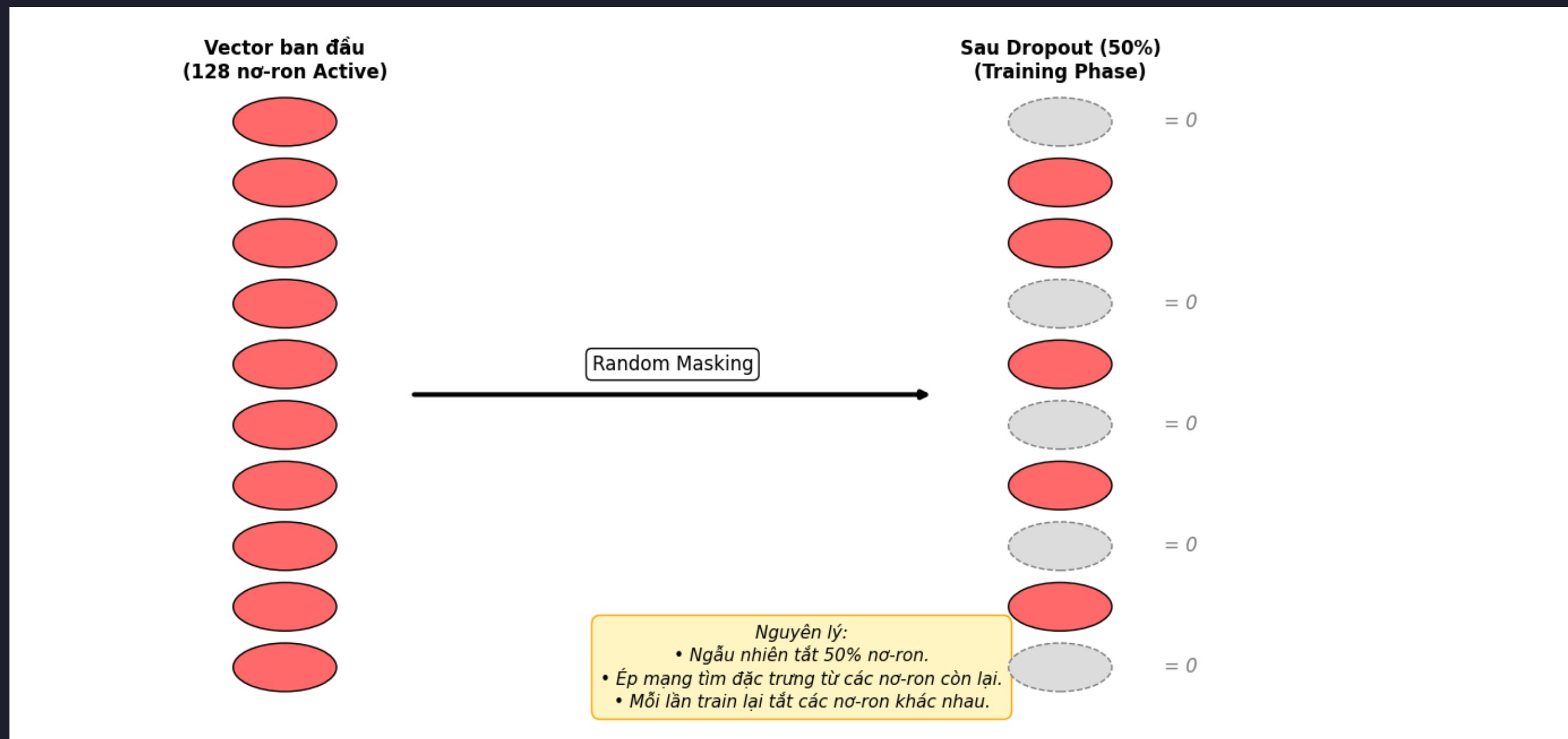
input: vector (128)

output: vector (128)

LAYER RELU & DROPOUT

ReLU: Lại lọc bỏ giá trị âm.

Dropout: (Chỉ chạy khi Training) Random tắt đi một số nơ-ron (gán bằng 0) với tỷ lệ 50%.



LAYER DENSE 2 (OUTPUT LAYER - CHẤM ĐIỂM)

Đây là điểm số thô mà mạng dành cho từng ký tự. Điểm càng cao, mạng càng nghi ngờ ảnh là ký tự đó.

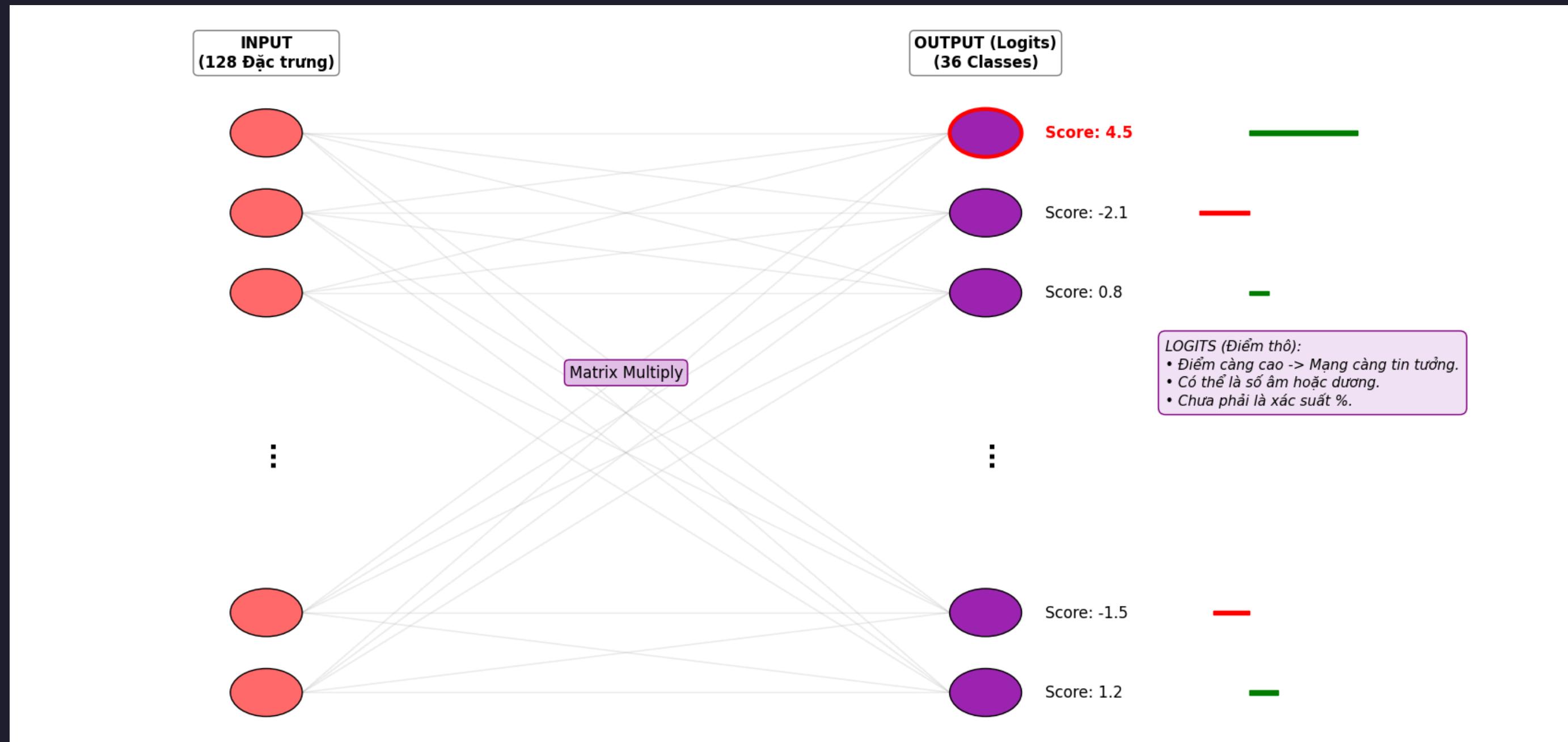
input: vector (128)

output: Vector có độ dài bằng số class (vector logits)

Ví dụ: [2.5, -1.0, 0.3, ...]

LAYER DENSE 2 (OUTPUT LAYER - CHẤM ĐIỂM)

Nhân ma trận để chuyển từ 128 đặc điểm sang số lượng Class (ví dụ: 36 class cho A-Z, 0-9).



LAYER SOFTMAX (XÁC SUẤT - KẾT LUẬN)

Trả lời câu hỏi cuối cùng: "Tôi tin 80% đây là chữ A, 10% là chữ B...".

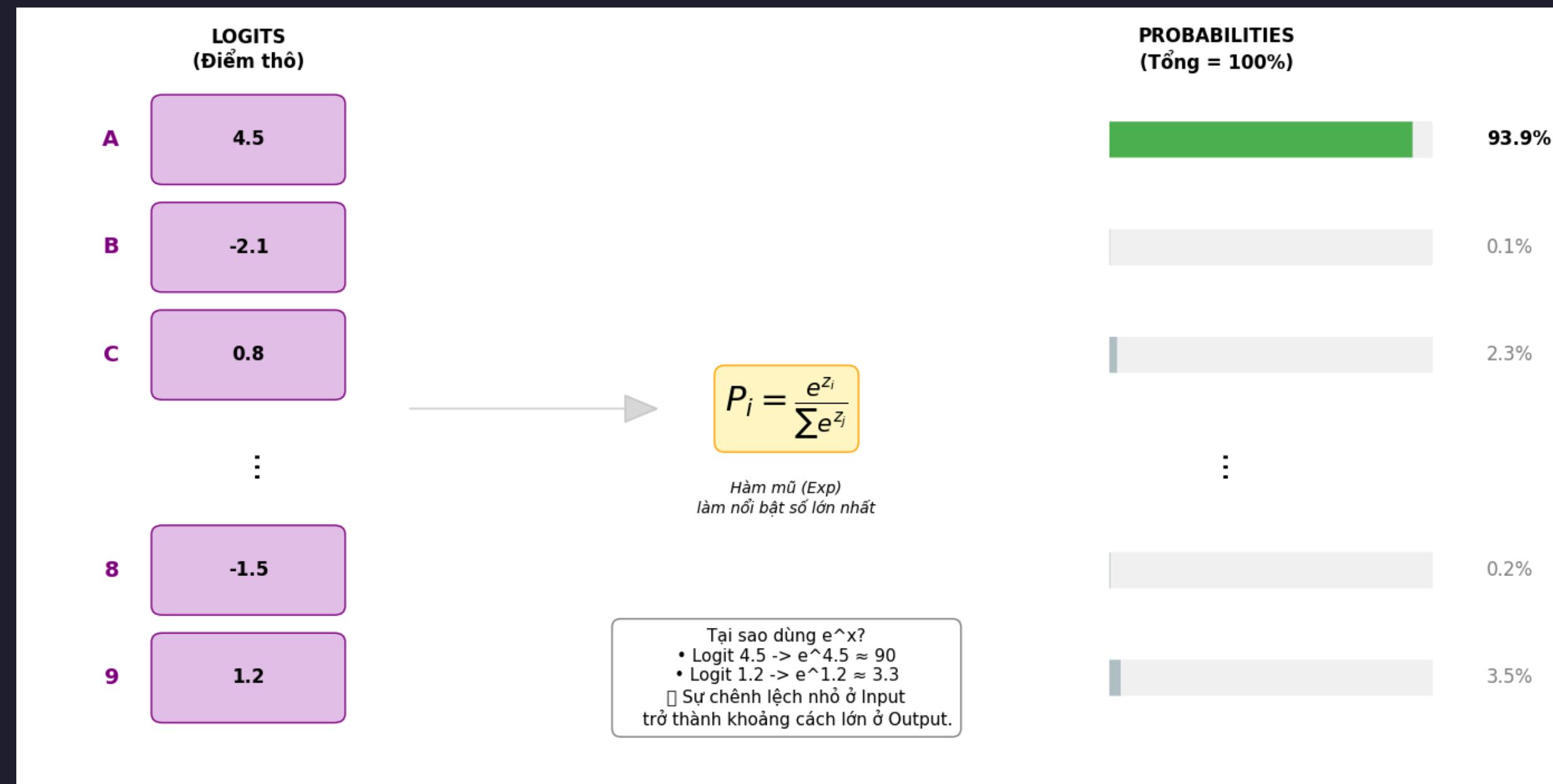
input: Vector Logits

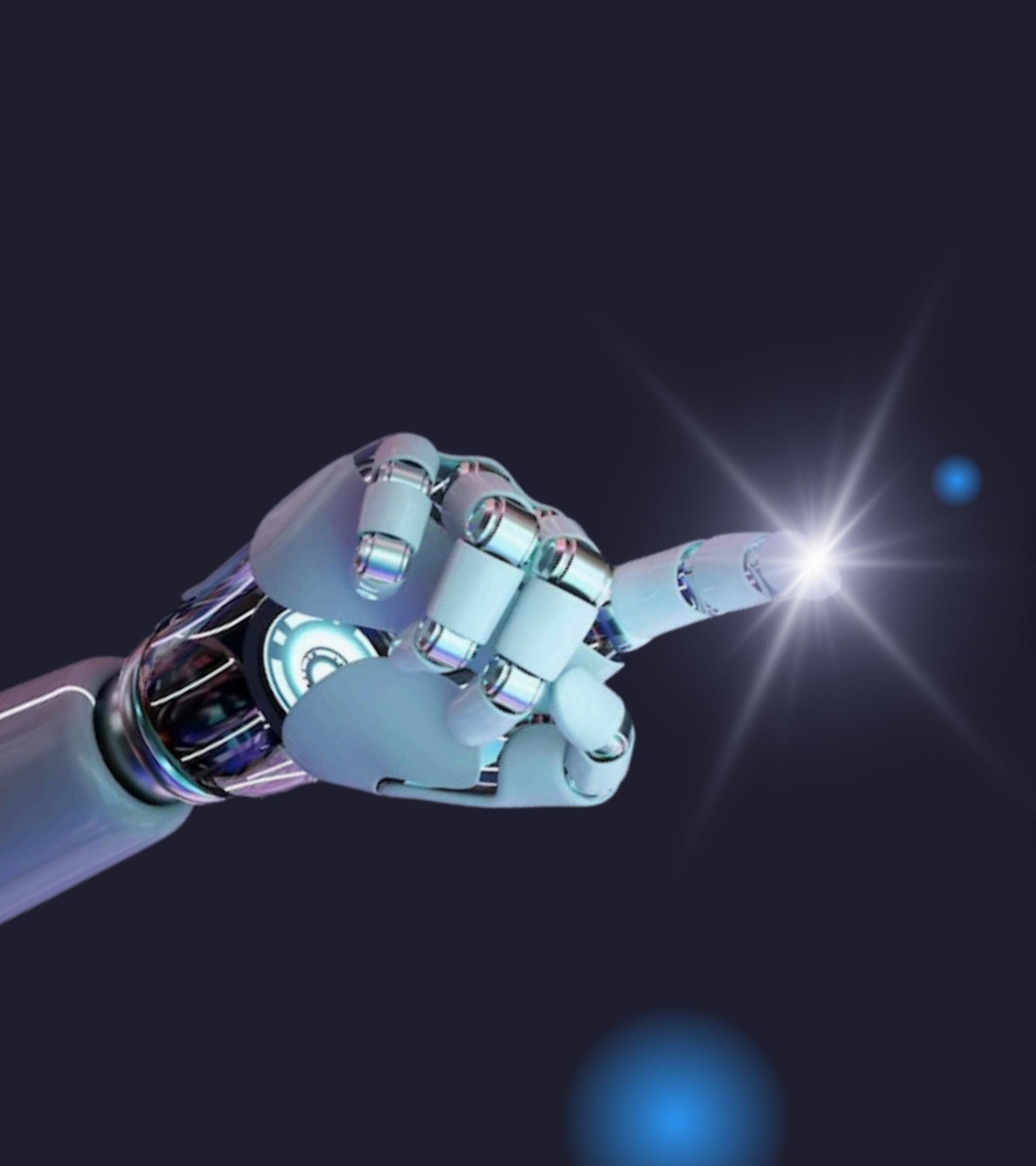
output: Vector xác suất có độ dài bằng số class (Probabilities).

Ví dụ: [0.8, 0.1, 0.05, ...]

LAYER SOFTMAX (XÁC SUẤT - KẾT LUẬN)

Dùng công thức hàm mũ để biến đổi điểm số thành phần trăm.





BACKWARD PROPAGATION

SỬA SAI KIẾN THỨC

LOSS CALCULATION

Trước khi sửa sai, máy phải biết mình sai trầm trọng đến mức nào.

input:

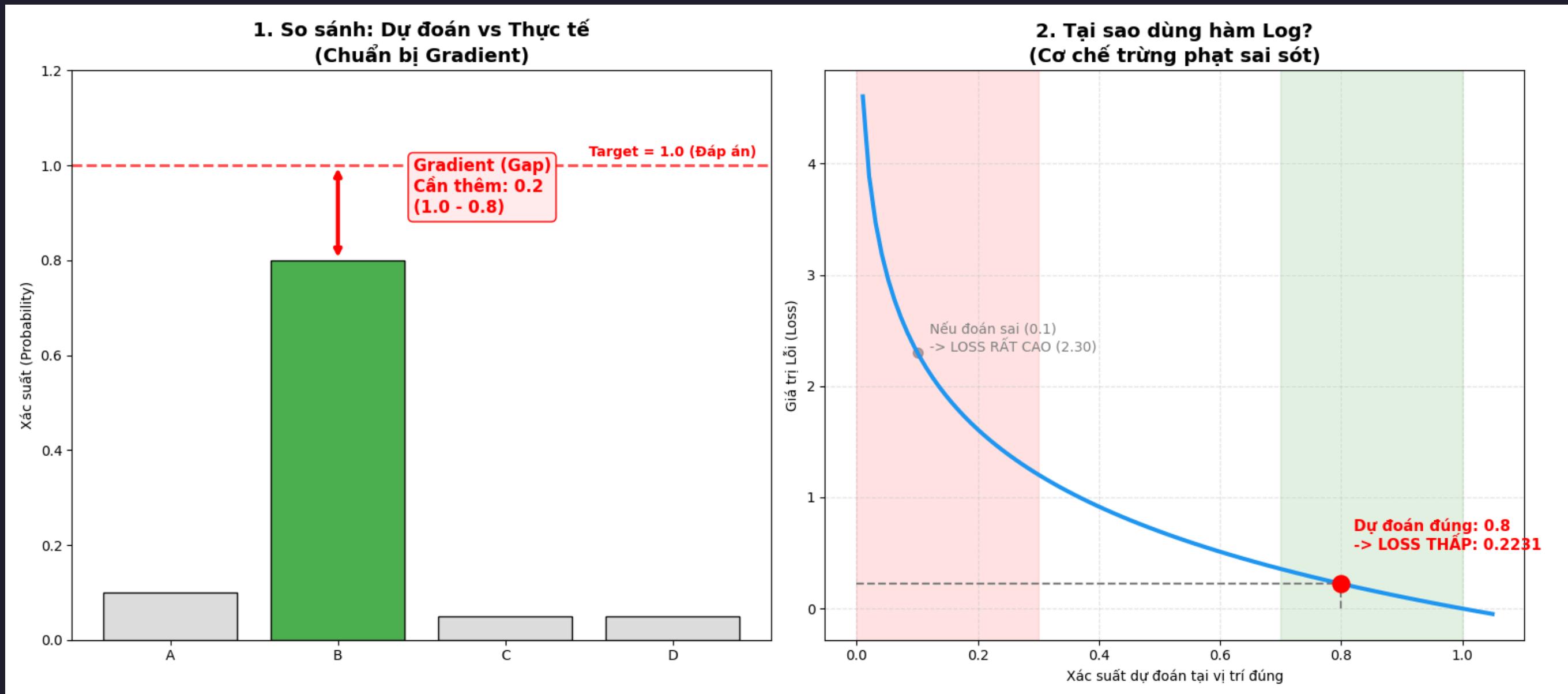
- probs: Vector xác suất từ Forward (ví dụ: [0.1, 0.8, 0.1, ...]).
- lbl: Đáp án đúng (ví dụ: số 1 tương ứng với chữ 'B').

output: gradient của loss

LOSS CALCULATION

Tính toán loss:

```
loss = -np.log(probs[lbl])
```



BACKWARD PROPAGATION



KẾT QUẢ TRAINING

