Problem 3e: Backward Check Explanation

Finite Difference Method for Gradient Verification

The `backward_check_dw` function verifies that the analytical gradient computed by `backward()` matches a numerical estimate using the finite difference method.

The Finite Difference Formula

To estimate the derivative numerically:

$$\frac{\partial h_k}{\partial W_{ij}} \approx \frac{h_k(W+\epsilon) - h_k(W-\epsilon)}{2\epsilon}$$

where ϵ is a small perturbation matrix with a single non-zero entry at position (i, j).

The Three Steps in Code

1. `self.w += e` $\rightarrow W \rightarrow W + \epsilon$
2. `self.w -= 2*e` $\rightarrow W + \epsilon \rightarrow W - \epsilon$ (subtract 2ϵ)
3. `self.w += e` $\rightarrow W - \epsilon \rightarrow W$ (restore original)

Variable Definitions

- **k**: Output index (0-9), which of the 10 output neurons we're checking
- **dh**: Upstream gradient, a one-hot vector with 1 at position k, e.g., $[0, 0, 0, 1, 0, 0, 0, 0, 0, 0]$ if $k = 3$
- **hm**: Input matrix of shape (1, 784), a random input image (batch_size=1)
- **e**: Perturbation matrix of shape (10, 784), all zeros except one small random value at position (i, j)
- **i**: Row index (0-9), which output neuron
- **j**: Column index (0-783), which input feature

Why This Sequence Works

After operation	Value of self.w
Start	W
`+= e`	$W + \epsilon$
`-= 2*e`	$W - \epsilon$
`+= e`	W (restored)

This efficiently computes both $f(W+\epsilon)$ and $f(W-\epsilon)$ while restoring W to its original value at the end.

Verification

The assertion `assert(np.linalg.norm(dw_e - dw[i, j]) < 1e-6)` checks that the numerical estimate `dw_e` matches the analytical gradient `dw[i, j]` within a small tolerance (10^{-6}).

If all assertions pass, the `backward()` implementation is correct.