

## Motivation

We study the paper *Mamba: Linear-Time Sequence Modeling with Selective State Spaces*. Our goal is to reimplement the core Mamba block from scratch, then extend it to spatial data using a bidirectional scan. We evaluate on datasets not used in the original paper to validate generalization. No custom CUDA kernels are used.

## Contributions

Clean PyTorch reimplementation of S6 selective scan (JIT) without custom CUDA. Vision extension via bidirectional scan over patch tokens. Evaluation scripts for MNIST, CIFAR-10, TinyShakespeare, plus baselines.

## Method Overview

Mamba replaces attention with a selective state space model (S6) that is input-dependent. We implement the recurrence with learned discretization  $\Delta$  and depth-wise causal convolution, followed by gating and projection.

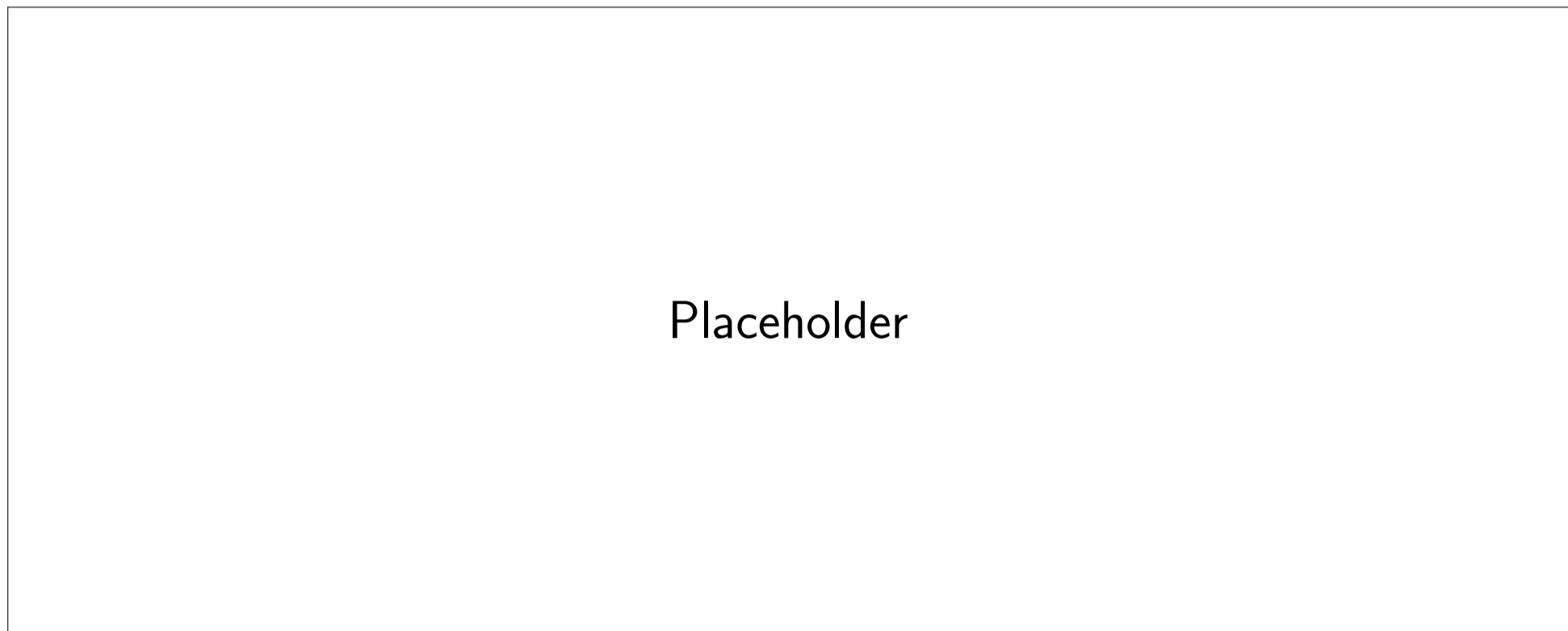


Figure 1: Mamba block diagram (placeholder).

## Selective Scan (S6)

For each time step  $t$ :

$$h_t = \Delta A \odot h_{t-1} + \Delta B \odot x_t, \quad y_t = C \odot h_t + D \odot x_t$$

where  $\Delta$  is input-dependent. We use a low-rank  $\Delta$  projection with S4D-style initialization for stability.

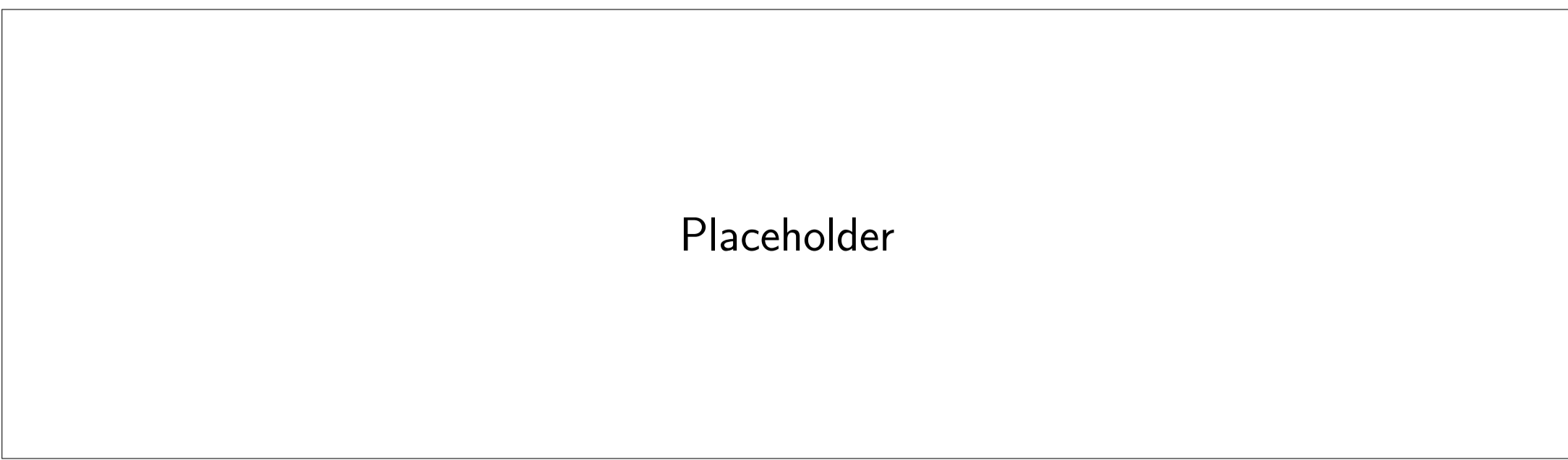


Figure 2: Selective scan illustration (placeholder).

## Vision Extension

For images, a single causal scan limits spatial context. We scan sequences in both forward and backward directions, then fuse the outputs. Input images are converted into  $4 \times 4$  patch tokens.

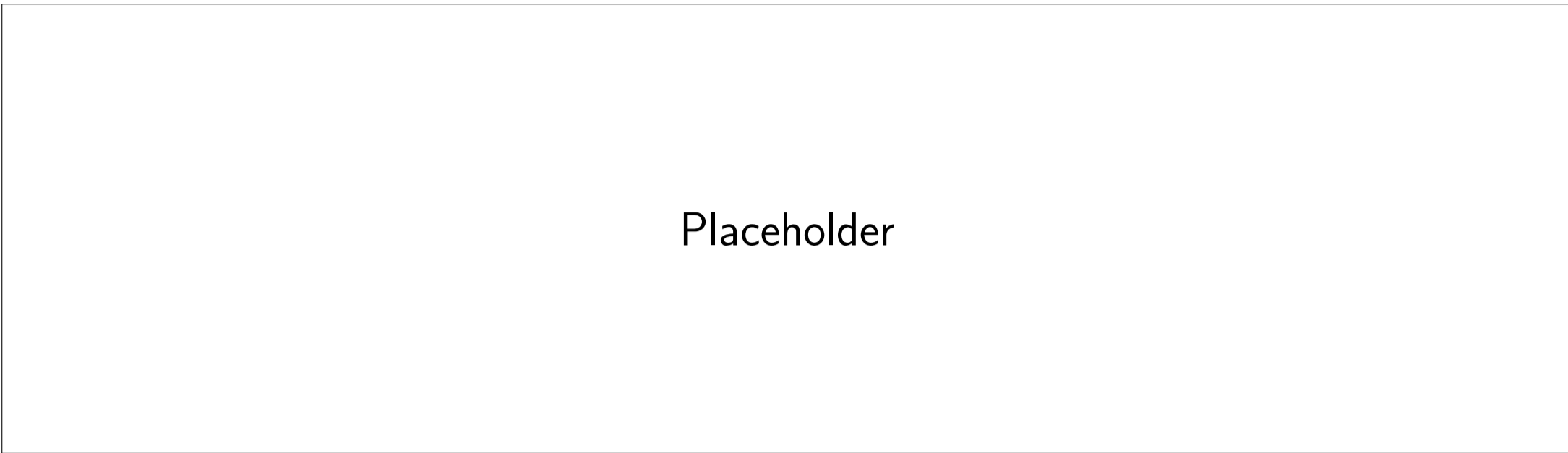


Figure 3: Bidirectional scan on image patches (placeholder).

## Implementation Details

Pure PyTorch implementation using `torch.jit.script`. Depthwise 1D convolution with kernel size  $d_{conv} = 4$ . S6 uses input-dependent  $\Delta$  with low-rank projection. RMSNorm at model output; residual connections per layer. Weight tying between token embedding and output head.

## Datasets and Evaluation

MNIST: Vision Mamba, causal ablation, and vanilla RNN baseline. CIFAR-10: patch-based image classification. TinyShakespeare: character-level language modeling.

Dataset	Key training settings (from train *.py)
MNIST (Vision)	d.model=64, layers=2, batch=64, lr=1e-3, epochs=5
MNIST (Causal)	d.model=64, layers=2, batch=64, lr=3e-4, epochs=5
MNIST (RNN)	input=16, hidden=64, layers=2, batch=64, lr=1e-3, epochs=5
CIFAR-10	d.model=128, layers=4, batch=64, lr=1e-3, epochs=5
TinyShakespeare	d.model=128, layers=4, block=128, batch=32, lr=1e-3, epochs=10

Table 1: Training settings summary.

## Results

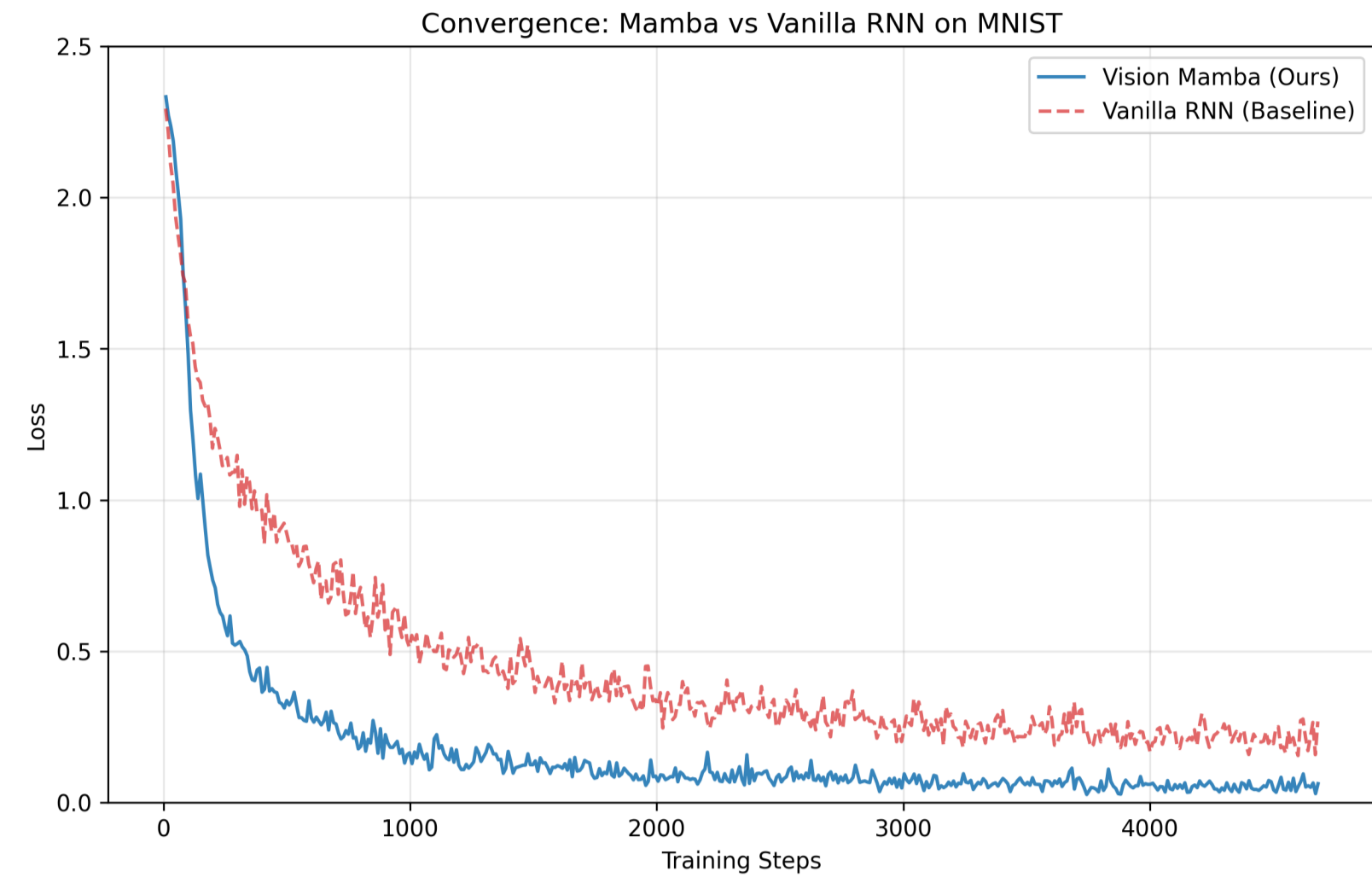


Figure 4: Vision Mamba vs RNN training loss (from outputs/figures).

Model	MNIST test accuracy (%)
Vanilla RNN (README)	94.40
Causal Vision Mamba (README)	97.00
Bi-Directional Vision Mamba (README / outputs)	97.93

Table 2: MNIST accuracy summary reported in README.md.

TinyShakespeare loss: min 1.1920 (step 5000), last 1.2393 (step 5300) from outputs/results/shakespeare/shakespeare\_results.csv.

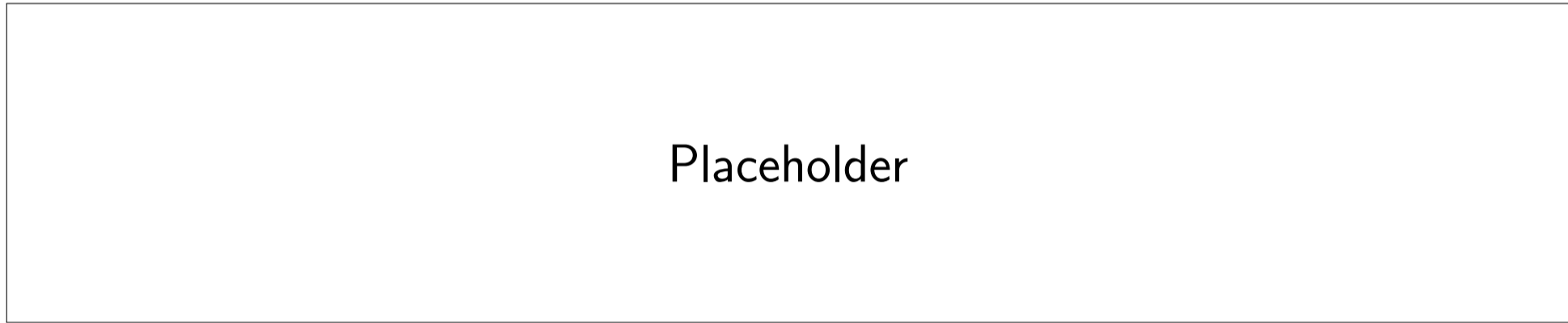


Figure 5: CIFAR-10 results (placeholder; no figure in outputs/).

## Ablation and Analysis

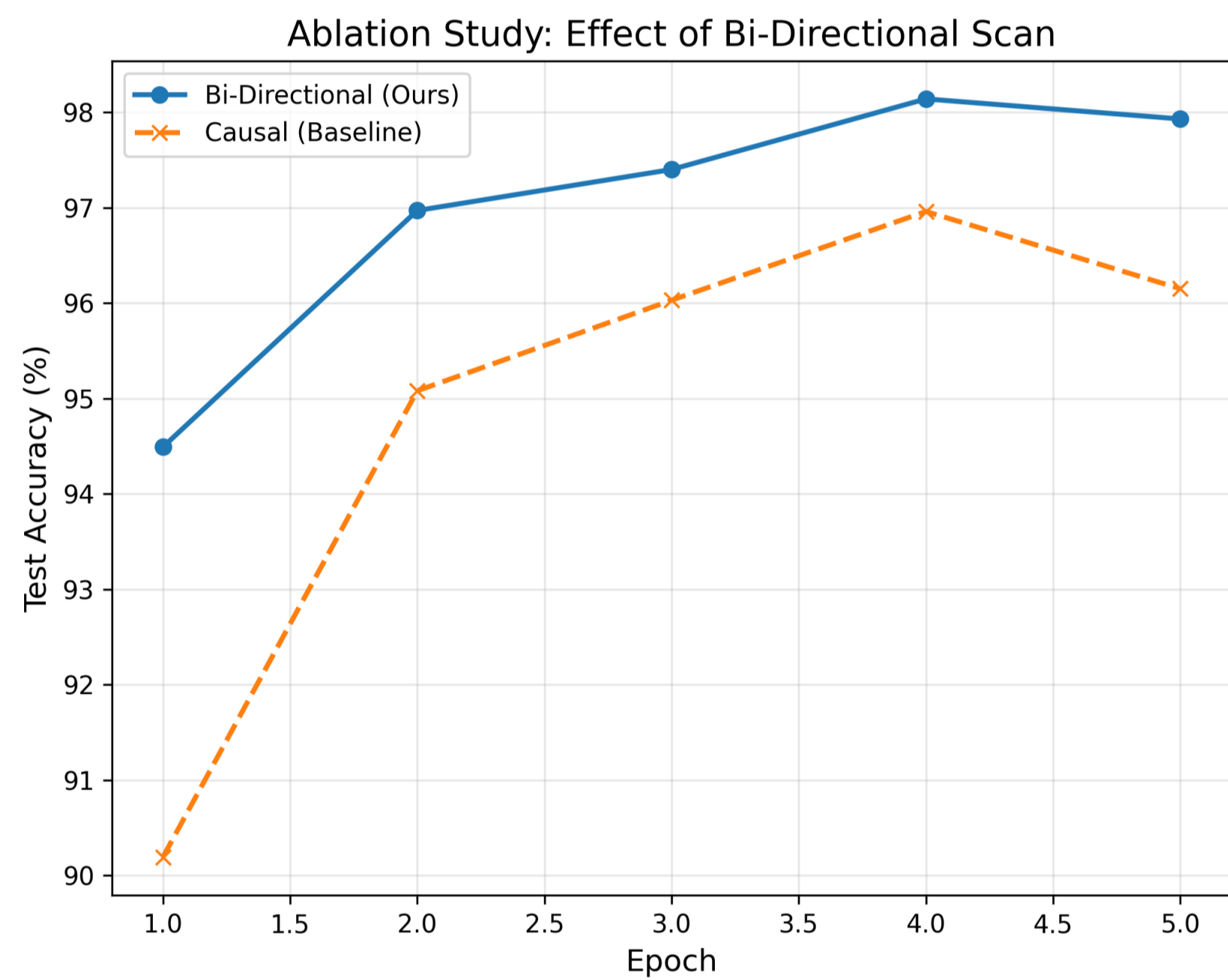


Figure 6: Ablation: causal vs bidirectional scan (from outputs/figures).

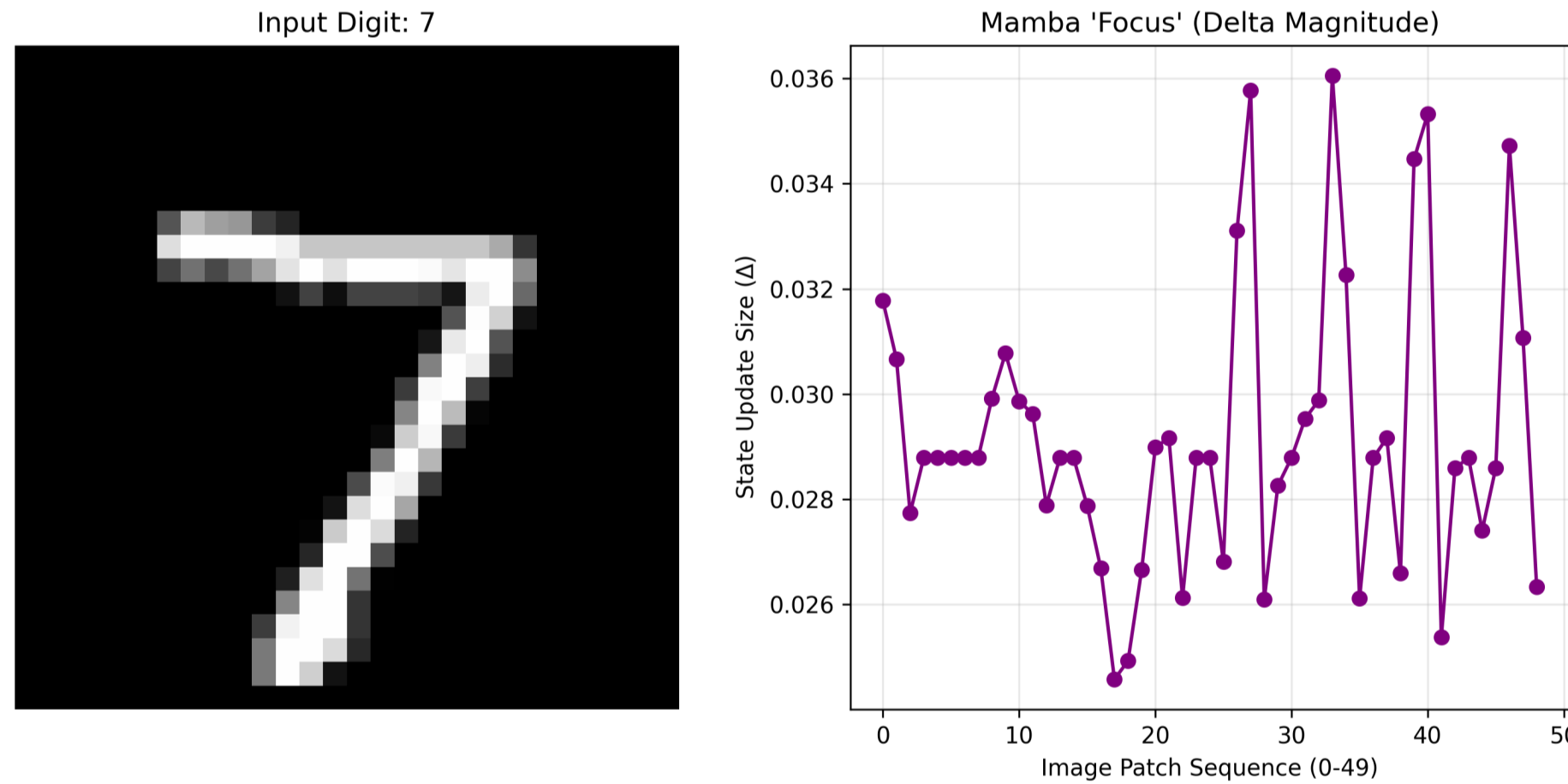


Figure 7: Visualization of  $\Delta$  (from outputs/figures).

## Conclusion and Future Work

Reimplemented core Mamba S6 and validated on new datasets. Bidirectional scan improves vision tasks. Future: larger-scale datasets and optimized kernels.

## References

Gu and Dao, *Mamba: Linear-Time Sequence Modeling with Selective State Spaces*, 2023.