Preserving Plasticity in Continual Learning with Adaptive Linearity Injection

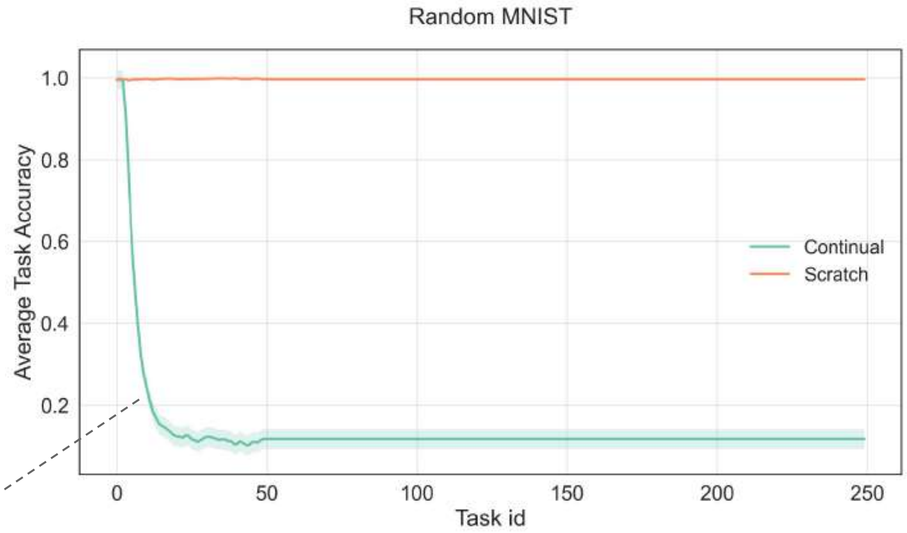
Seyed Roozbeh Razavi Rohani[1*], **Khashayar Khajavi**[1*], Wesley Chung[2,3], Mo Chen[1] and Sharan Vaswani[1]

CoLLAs 2025
Presented at **AI/CRV 2026**

1 Simon Fraser University, 2 McGill University, 3 Mila
* Equal Contribution

Loss of Plasticity: When Models Stop Learning

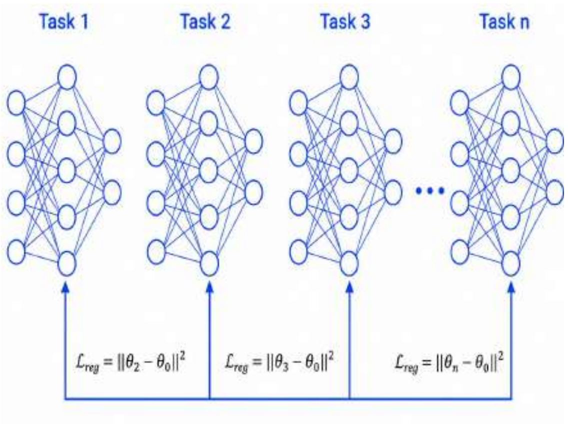
- **Plasticity:** the ability of a model to adapt to new data
- Important in **Continual Learning** and **Reinforcement Learning**
- **Loss of plasticity:** reduced ability to learn as training progresses



Continual training collapses, while training from scratch succeeds!

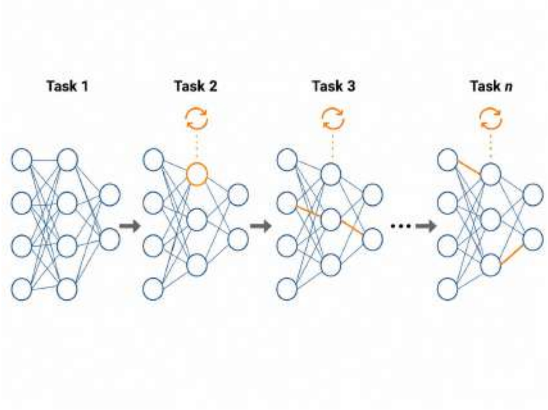
Existing Approaches and a Different Direction

Regularization
(Kumar et al., 2023, Chung et al., 2025)



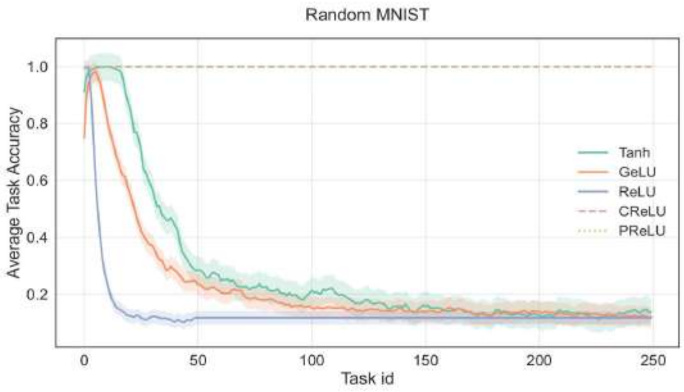
Can limit expressivity

Resetting
(Dohare et al., 2024)



Requires task boundaries

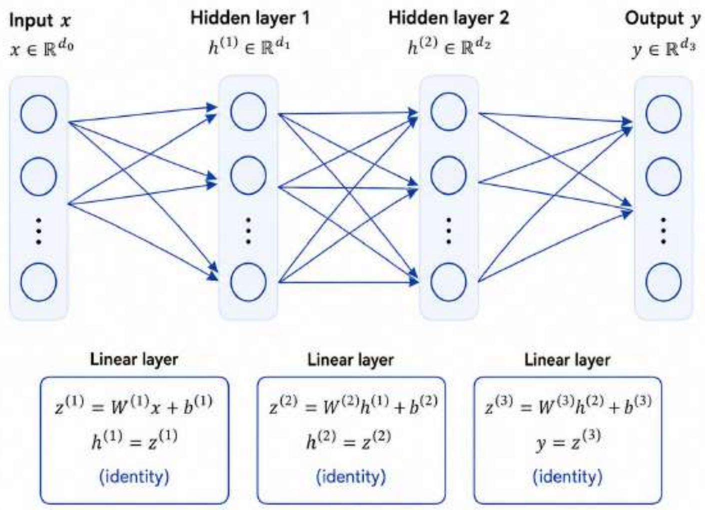
Our Direction



Some activations preserve plasticity

Motivation: Linearity Helps Plasticity

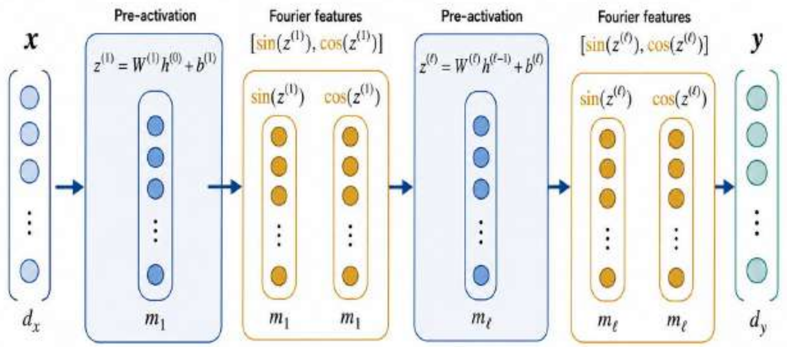
Deep Linear Networks



✓ preserve plasticity
✗ limited expressivity

Deep Fourier Features

(Lewandowski et al. 2025)



✓ combine linearity and nonlinearity
✗ fixed balance: always outputs (sine, cosine)

Can we adaptively balance linearity and nonlinearity?

For each neuron, we aim to adaptively inject linearity into its activation function

Our Approach: Adaptive Linearization (AdaLin)

We augment any non-linear activation function ϕ with a Lipschitz constant L with a learnable linear term:

$$f_i(x) = \phi(x) + \alpha_i . x . [g(x)]_{sg}$$

Output of Neuron i

Preactivation

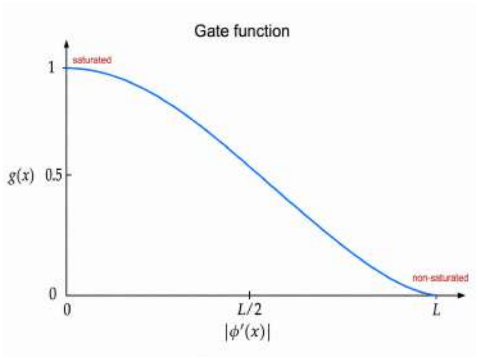
Learnable Parameter
Per-neuron

where:

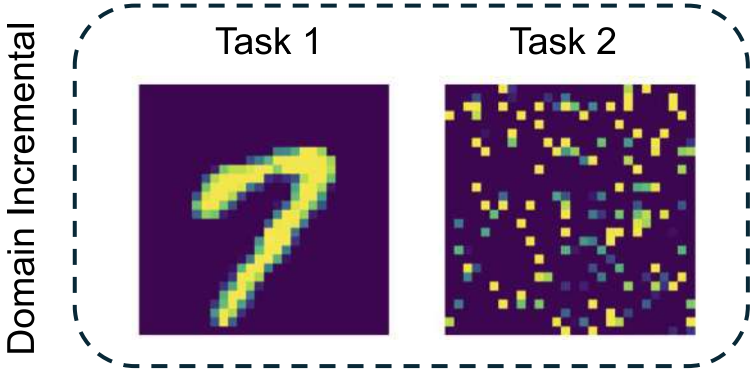
$$g(x) = cos(\frac{\pi}{2} . \frac{|\phi'(x)|}{L})$$

Small $|\phi'(x)|$ —> Larger Gate —> More Linearity

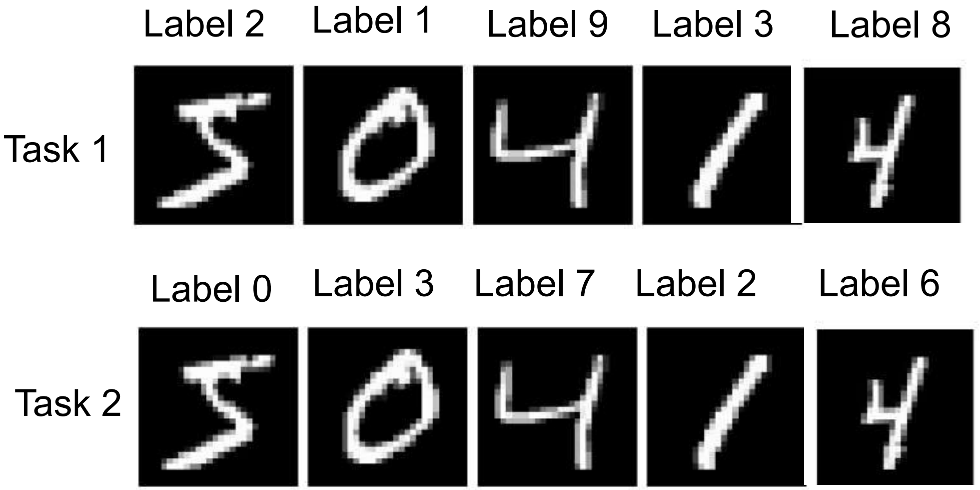
Large $|\phi'(x)|$ —> Smaller Gate —> More Non-Linearity

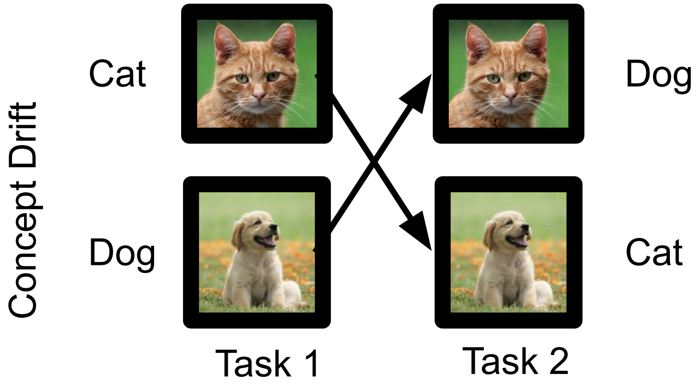


Experiments – Benchmarks

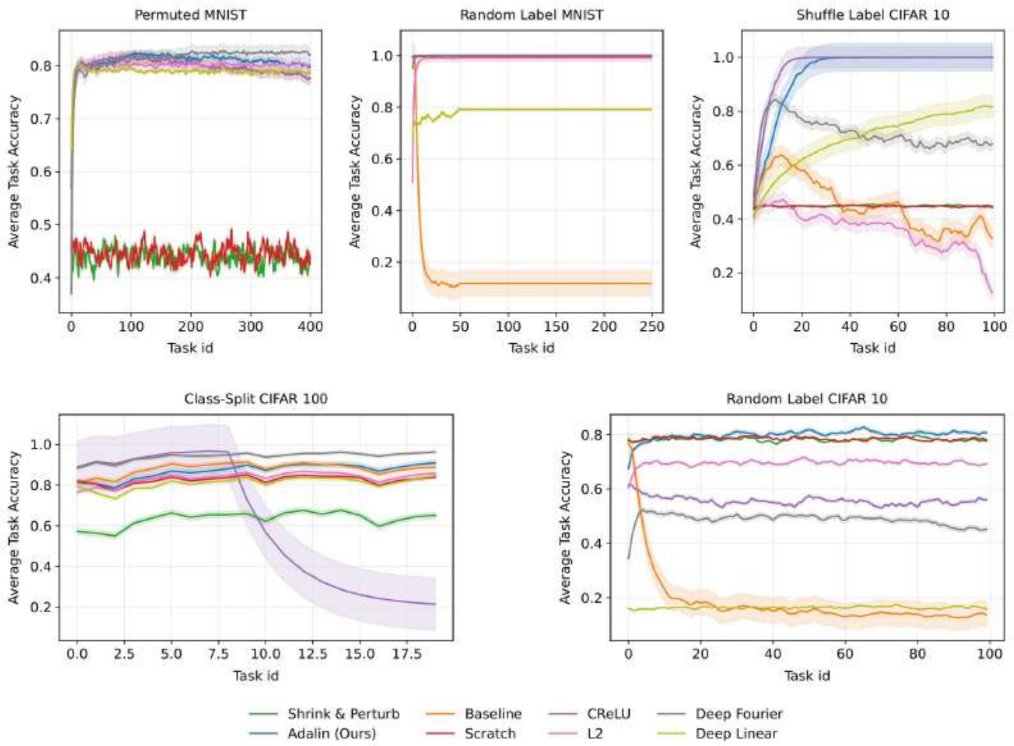


Memorization

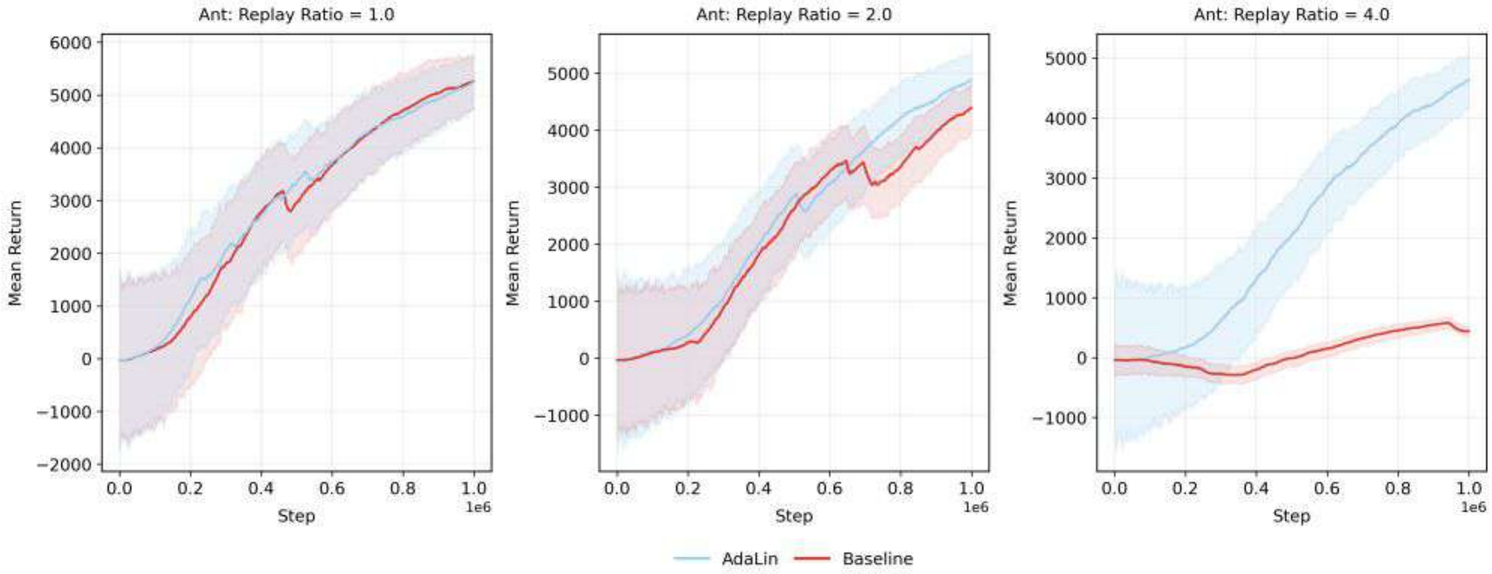




Experiments – Supervised Learning

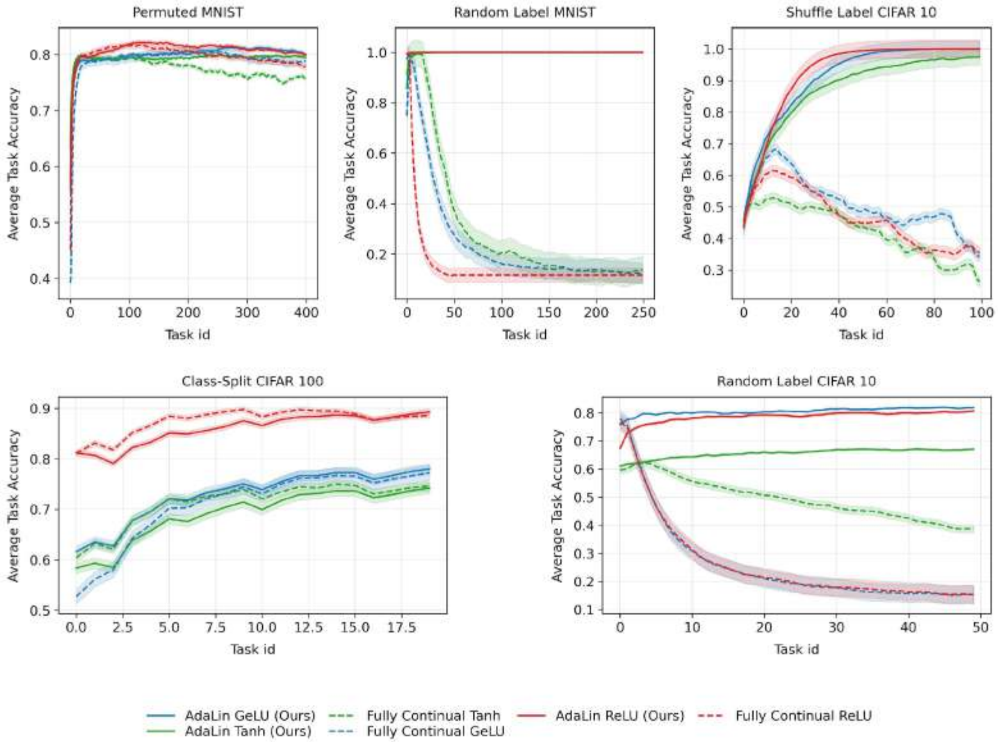


Experiments – RL



Ablations

- AdaLin can be applied to different activation functions



Thank You!