# Ablation of Gate Biases in GRUs for s-CIFAR-10

Kieran A. Malandain
Department of Physics
Yale University
New Haven, CT, USA
kieran.malandain@yale.edu
Study completed as part of the application process for Great Sky Inc.
Code and further documentation can be found on this GitHub repo
February 2026

***Abstract*—This short study explores the effect of learnable update and reset biases when using a GRU RNN for s-CIFAR-10 (sequentialized) classification. We do not observe significant performance differences between the baseline (learnable) and ablated (fixed biases) models, suggesting that the ablated model is able to simulate the effect of learnable biases through the weight matrices.**

## I. Introduction

Processing sequential representations of high-dimensional spatial data requires recurrent architectures capable of retaining information through an extended history. This study explores the performance of Gated Recurrent Units (GRUs) on the sequentialized CIFAR-10 dataset, taken in grayscale. Typically, computer vision (CV) relies on convolutional neural networks (CNNs), which use kernel optimization to learn, for example, image classification. Here, we take the $32 \times 32$ images in the CIFAR-10 dataset and flatten them into a 1024-step sequence, instead classifying them with a recurrent neural network (RNN). This acts as a benchmark of the backpropagation through time (BPTT) mechanism. Current literature demonstrates that specialized networks, e.g., state-space models (SSMs) or residual networks (ResNets), can achieve $\sim 90\%$ accuracy on this task, vanilla RNNs like the GRU-based models we employ here inherently plateau in the $\sim 60\%$ range, even with extensive hyperparameter tuning[1] [2], [3].

The primary method for memory retention in a vanilla GRU is the update gate $z_t$. The hidden state update rule:

$$h_t = (1 - z_t) \odot n_t + z_t \odot h_{(t-1)}, \tag{1}$$

[1]Note that improving the gating mechanism can boost GRU accuracy on s-CIFAR-10 to around $\sim 70\%$ [1], but here we are only interested in more vanilla approaches.

requires $z_t \to 1$ to preserve the context over long horizons (prioritizing the previous hidden state $h_{t-1}$ over the candidate new state $n_t$). The reset gate $r_t$ dictates how much past context is mixed into the new candidate state $n_t$. In this study, we investigate the power of the learnable update bias $b_z$ and reset bias $b_r$ in retaining long-term memory and preventing catastrophic forgetting. We run two models: (1) a baseline, where all biases (update $b_z$, reset $b_r$, and candidate $b_n$) are learnable parameters; and (2) an ablated, where we block the network from learning $b_z$ and $b_r$ and instead keep them at a value of `0.0`. This forces $z_t \approx \sigma(0) = 0.5$, where $\sigma(\cdot)$ is the sigmoid function. Over the $T = 1024$ steps of s-CIFAR-10, a continuous 50% decay can easily result in catastrophic forgetting.

## II. Methodology & Implementation

Both models (baseline and ablated) that we compare are trained with the same underlying RNN structure and hyperparameters. The only difference lies in that the baseline model allows *all* biases to be learned throughout training, whereas the ablated model *fixes* the update and reset biases $b_z$ and $b_r$ to `0.0`, whilst still allowing the candidate bias $b_n$ to be learned.

### A. Architectural Constraints

We are constrained to using a model with around 100K parameters, and as such we train a single-layer GRU with a hidden dimension of $H = 181$ with a 1D (grayscale) input; in total, this yields 101,732 trainable parameters. We need to be aware of gradient problems with BPTT, and so we initialized the hidden-to-hidden weight matrices $W_{hh}$ as orthogonal, which ensures that they have initial eigenvalues of magnitude 1, preserving gradient magnitude.

### B. Optimization

Initial training runs revealed that a fixed learning rate in the range of $1e^{-3}$ was too high to see convergence, and so we implemented a slightly lower base learning rate of $5e^{-4}$ together with a Cosine Annealing scheduler to bring this down (across 40 epochs) to a minimum of $1e^{-5}$. For an optimizer, we use `AdamW`, which has significant ($\sim 15\%$) improvements over Adam [4], applying a weight decay. Decoupling the weight decay from the adaptive momentum (the difference between `Adam` and `AdamW`) prevents the corruption of moving averages, allowing us to get a stable convergence with a lower risk of BPTT cliff-jumping.

### C. Baseline Model

For the baseline model, we attempted to implement the chrono initialization per Tallec and Olliver (2018) [5], wherein we sample the biases $b \sim \log(\mathcal{U}([1, T_{\max} - 1]))$. Empirical tests of this demonstrate that this initialization method actually severely degraded performance compared to initializing the update bias uniformly to `+1.0`, decreasing prediction accuracy by approximately 25 percentage points. The reason for this is that the GRU update gate is coupled in the sense that it controls both how much of the previous hidden state $h_{t-1}$ and the new candidate state $n_t$ is included in the new hidden state $h_t$ (see (1)). The expected value of the bias under this distribution is about `+6.2`. This pushes the sigmoid to $z_t = \sigma(6.2) \approx 0.998$, and therefore the network essentially ignores any new inputs. The `+1.0` uniform initialization is a better choice for preserving BPTT gradients while maintaining input sensitivity, giving $z_t \approx 0.73$.

### D. Ablated Model

For the ablated model, we implemented backward hooks in PyTorch to isolate the update and reset biases without sacrificing the execution speed native to PyTorch (via fused cuDNN kernels). During the backward pass, this hook zeros the gradient update of the update and reset biases, thus keeping them at their original value—in the ablated model, `0.0`. The candidate biases remain able to learn.

## III. Results & Discussion

The model was trained for 40 epochs on an NVIDIA RTX 5000. Both models achieved a final test accuracy of $\sim 50\%$ (fig. 1), and with additional training could have further converged (fig. 2) towards the $\sim 60\%$ vanilla RNN limit [3]. The training dynamics shown in the figures illustrate the differences that the biases can make.

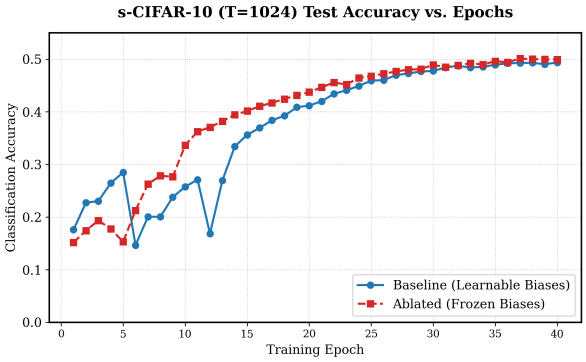


Fig. 1. Test accuracy of both models

- **Forced memory (baseline):** Due to our bias initialization, the net was forced into a high-retention state from the beginning. This allowed early gradient flow across a higher number of steps; however, it gave a more volatile loss landscape. We see in both figures that this resulted in a BPTT cliff at epoch 6, where the weights are shattered by the large update.[2] This caused a severe spike in the training loss (fig. 2) and a decrease in test accuracy (fig. 1).
- **Overparameterization (ablated):** The ablated model initially failed to learn significantly, with the training loss remaining high and fairly constant for the first five epochs. Plausibly, this is due to forcing the update and reset biases to remain at `0.0`, meaning the net has a difficult time retaining significant information across time steps (due to vanishing gradients). Ultimately, however, this actually shielded the ablated model from the same BPTT cliff that the baseline model encountered, and so the optimizer can slowly inflate the parameter magnitudes within the $W_{hz}$ weight matrix to *simulate* a positive bias offset. This acted as a natural curriculum and explains why the final performance was similar across both models.

That the ablated model was able to perform almost identically to the baseline model with learnable parameters demonstrates that the network can mathematically route around the missing bias, inflating weight matrices to organically simulate a memory curriculum. However,

[2]Note that this BPTT cliff occurred despite the application of gradient clipping ($L_2$ norm clamped to 1.0) during training, underscoring the non-convexity of the loss landscape. Stricter clipping could further mitigate this.

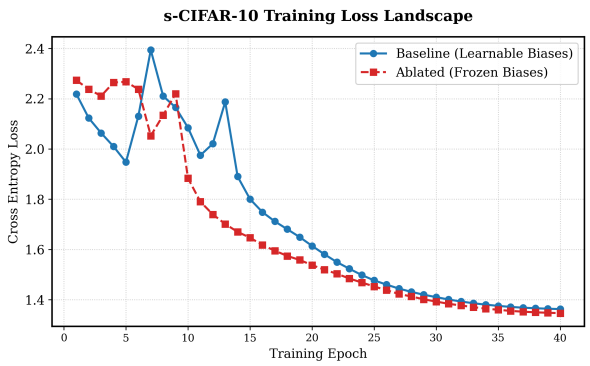


Fig. 2. Training loss of both models

this is not as simple where the network is physically instantiated in the hardware, e.g., in Superconducting Optoelectronic Networks (SOENs). Here, a temporal bias corresponds to an injected current setting a fixed, physical time constant—e.g., the decay rate of a fluxon loop. Inflating a physical synaptic routing weight here cannot be done without some physical effect on the system, which could affect spatial and cryogenic constraints and cause the network to physically fail. The early instability of the baseline model highlights the optimization challenge on continuous-time hardware.[3]

## IV. Conclusion

To conclude this short study, we have explored the performance of two GRU RNNs on the s-CIFAR-10 task: one baseline model where all biases are fully learnable, and the update and reset initialized to `+1.0`, and one ablated model where the update and reset biases are not learnable parameters and instead fixed at `0.0`. We observed that ablation affects early training but, over 40 epochs, there is no significant difference in model performance. We conclude that this is due to the ablated network simulating positive biasing through slow updates of the weight matrix $W_{hz}$. Finally, we note that these dynamic, simulation-based updates are not as simple to implement when the network is physically instantiated in the hardware, e.g., in SOENs.

## References

[1] A. Gu, C. Gulcehre, T. L. Paine, M. Hoffman, and R. Pascanu, "Improving the Gating Mechanism of Recurrent Neural Networks," 2019. [Online]. Available: https://arxiv.org/abs/1910.09890

[2] B. Chang, M. Chen, E. Haber, and E. H. Chi, "AntisymmetricRNN: A Dynamical System View on Recurrent Neural Networks," 2019. [Online]. Available: https://arxiv.org/abs/1902.09689

[3] A. Gu, K. Goel, and C. Ré, "Efficiently Modeling Long Sequences with Structured State Spaces," 2021. [Online]. Available: https://arxiv.org/abs/2111.00396

[4] I. Loshchilov and F. Hutter, "Decoupled Weight Decay Regularization," 2017. [Online]. Available: https://arxiv.org/abs/1711.05101

[5] C. Tallec and Y. Ollivier, "Can recurrent neural networks warp time?" 2018. [Online]. Available: https://arxiv.org/abs/1804.11188

[6] R. O'Loughlin, B. Primavera, and J. Shainline, "Dendritic Learning in Superconducting Optoelectronic Networks," in *Proceedings of the 2023 International Conference on Neuromorphic Systems*, ser. ICONS '23. ACM, Aug. 2023, p. 1–8. [Online]. Available: http://dx.doi.org/10.1145/3589737.3605972

[3] I believe Great Sky has worked on Elastic Weight Collision, see [6].