NEURAL ARITHMETIC UNITS

Andreas Madsen
Computationally Demanding
amwebdk@gmail.com

Alexander Rosenberg Johansen
Technical University of Denmark
aler@dtu.dk

ABSTRACT

Exact addition, subtraction, multiplication, and division of real numbers present a unique learning challenge for machine learning models. Neural networks can approximate complex functions by learning from labeled data. However, when extrapolating to out-of-distribution samples neural networks often fail. Learning the underlying logic, as opposed to an approximation, is crucial for applications such as comparing, counting, and inferring physical models. Our proposed Neural Addition Unit (NAU) and Neural Multiplication Unit (NMU) can learn the underlying rules of real number addition, subtraction, and multiplication by merely using backpropagation. The core novelty of NAU and NMU is their ability to efficiently learn sparse weights, which allows the unit to perform exact arithmetic operations. The NAU and NMU are inspired by the underlying arithmetic components of the Neural Arithmetic Logic Unit (NALU). Through analytic analysis supported by empirical evidence we justify how the NAU and NMU improve over the NALU and its underlying components. Our experiments include the an much expanded version of the arithmetic extrapolation task from the NALU paper, and a variant of the sequential MNIST task from the NALU paper where multiplication is used instead of addition. We show that NAU and NMU have fewer parameters, converges more consistently, learns faster, and have more meaningful discrete values than the NALU and its arithmetic components.[1]

1 INTRODUCTION

When studying intelligence, insects, reptiles, and humans have been found to possess neurons with the capacity to hold integers, real numbers, and perform arithmetic operations (**???**). In our quest to mimic intelligence we have put much faith in neural networks, which in turn has provided unparalleled and often superhuman performance in tasks requiring high cognitive abilities (**???**). However, when using neural networks to learn simple arithmetic problems, such as counting, multiplication, or comparison they systematically fail to extrapolate onto unseen ranges (**???**). This can be a significant drawback when comparing in question answering (**?**) or counting objects in visual data (**??**).

In this paper, we analyze and improve parts of the recently proposed Neural Arithmetic Logic Unit (NALU) (**?**), which we will introduce in section 2. Our contributions are; an alternative formulation of the soft weight constraint with a clipped linear activation, parameter regularization that biases towards a sparse solution of $\{-1, 0, 1\}$, and a reformulation of the multiplication unit with a partial linearity. All of which significantly improves upon the existing NAC_+ and $\text{NAC}_\bullet$ units as shown through extensive testing on static arithmetic tasks.

The NALU is a neural network layer with two sub-units; the NAC_+ for addition/subtraction and the $\text{NAC}_\bullet$ for multiplication/division. The sub-units are softly gated using a sigmoid function. The parameters, which are computed by a soft weight constraint using a tanh-sigmoid transformation, are learned by observing arithmetic input-output pairs and using backpropagation (**?**). We motivate our work by an investigation of the NALU components; the parameter transformation, the sub-units, NAC_+ and $\text{NAC}_\bullet$, and the soft gating mechanism. The investigation uncovers the following analytical and empirical concerns; the gradients of the weight matrix construction in NAC_+ and $\text{NAC}_\bullet$

[1]Implementation is available on GitHub: https://github.com/AndreasMadsen/stable-nalu.

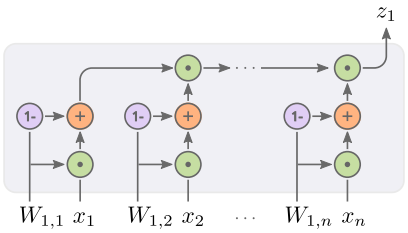


Figure 1: Visualization of NMU for a single output scalar z_1, this construction repeats for every element in the output vector $\mathbf{z}$.

have zero expectation, the $\text{NAC}_\bullet$ has a treacherous optimization space with unwanted global minimas near singularities, when applying the NAC_+ in isolation we observe that the wanted weight matrix values of $\{-1, 0, 1\}$ are rarely found, and our empirical results reveals that the NALU is significantly worse than hard-choosing either the NAC_+ or $\text{NAC}_\bullet$, indicating that the gating mechanism does not work as intended.

We avoid using gating as we see no obvious solution to simultaneously train two vastly different sub-units, the NAU/NAC_+ and NMU/$\text{NAC}_\bullet$, with a soft gating mechanism. We will thus assume that the desired operation is already known, or can empirically be found by varying the network architecture.

1.1 Learning a 10 parameter function

Consider the static function $t = (x_1 + x_2) \cdot (x_1 + x_2 + x_3 + x_4)$ for $x \in \mathbb{R}^4$. To illustrate the ability of $\text{NAC}_\bullet$, NALU, and our proposed NMU we attempt to fit this function by only observing input-output pairs.

For the $\text{NAC}_\bullet$, NALU, and NMU we conduct 100 experiments with different seeds and stop after $2 \cdot 10^5$ iterations. The results in table 1 show that NMU has a higher success rate and converges faster.

Table 1: Shows the success-rate for $\mathcal{L}_{\mathbf{W}_1,\mathbf{W}_2} < \mathcal{L}_{\mathbf{W}_1^\epsilon,\mathbf{W}_2^*}$, at what global step the model converged at and the sparsity error for all weight matrices, with 95% confidence interval. Best result is highlighted without considering significance.

Op	Model	Success	Solved at		Sparsity error
		Rate	Median	Mean	Mean
$\times$	$\text{NAC}_\bullet$	13%	$5.5 \cdot 10^4$	$5.9 \cdot 10^4 \pm 8.4 \cdot 10^3$	$7.5 \cdot 10^{-6} \pm 2.0 \cdot 10^{-6}$
	NALU	26%	$7.0 \cdot 10^4$	$7.6 \cdot 10^4 \pm 8.2 \cdot 10^3$	$9.2 \cdot 10^{-6} \pm 1.7 \cdot 10^{-6}$
	NMU	**94%**	$\mathbf{1.4 \cdot 10^4}$	$\mathbf{1.4 \cdot 10^4 \pm 2.2 \cdot 10^2}$	$\mathbf{2.6 \cdot 10^{-8} \pm 6.4 \cdot 10^{-9}}$

2 Introducing differentiable binary arithmetic operations

We define our problem as learning a set of static arithmetic operations between the elements of a vector. E.g. for a vector $\mathbf{x}$ learn the function $x_5 + x_1 \cdot x_7$. This is formalized as

$$z_{h_\ell} = z_1 \circ_{\ell-1} z_2 \circ_{\ell-1} \cdots z_{k-1} \circ_{\ell-1} z_k \mid z_1, z_2, \cdots, z_k \in \mathbf{z}_{\ell-1}, \circ_{\ell-1} \in \{+, -, \times, \div\}. \quad (1)$$

The Neural Arithmetic Logic Unit (NALU) (**?**) attempts to solve problem 1 by introducing two sub-units; the NAC_+ and $\text{NAC}_\bullet$ that exclusively represent either the $\{+, -\}$ or the $\{\times, \div\}$ operations. The NALU attempts to have either NAC_+ or $\text{NAC}_\bullet$ selected exclusively, which could require the NALU to be applied multiple times (alternating between NAC_+ and $\text{NAC}_\bullet$) in order to represent the entire space of solutions for problem 1.

The NAC_+ and $\text{NAC}_\bullet$ are defined accordingly,

$$W_{h_\ell,h_{\ell-1}} = \tanh(\hat{W}_{h_\ell,h_{\ell-1}})\sigma(\hat{M}_{h_\ell,h_{\ell-1}}) \tag{2}$$

$$\text{NAC}_+ : \; z_{h_\ell} = \sum_{h_{\ell-1}=1}^{H_{\ell-1}} W_{h_\ell,h_{\ell-1}} z_{h_{\ell-1}} \tag{3}$$

$$\text{NAC}_\bullet : \; z_{h_\ell} = \exp\left(\sum_{h_{\ell-1}=1}^{H_{\ell-1}} W_{h_\ell,h_{\ell-1}} \log(|z_{h_{\ell-1}}| + \epsilon)\right) \tag{4}$$

where $\hat{\mathbf{W}}, \hat{\mathbf{M}} \in \mathbb{R}^{H_\ell \times H_{\ell-1}}$ are trainable weight matrices, $z_{h_{\ell-1}}$ is the input of size $H_{\ell-1}$, and H_ℓ is the output size. The matrices are combined using a tanh-sigmoid transformation to bias the parameters towards a $\{-1, 0, 1\}$ solution. Having $\{-1, 0, 1\}$ allows a linear layer to exactly emulate the binary $\{+, -\}$ operation between elements of a vector as used when computing the NAC_+. The $\text{NAC}_\bullet$ extends the NAC_+ by using an exponential-log transformation, which, with $\{-1, 0, 1\}$ weight values, becomes the $\{\times, \div\}$ operations (within ϵ precision).

The NALU combines these units with a gating mechanism $\mathbf{z} = \mathbf{g} \odot \text{NAC}_+ + (1 - \mathbf{g}) \odot \text{NAC}_\bullet$ given $\mathbf{g} = \sigma(\mathbf{G}\mathbf{x})$. The idea is that the NALU should be a plug-and-play component in a neural network that has the ability to, with stochastic gradient descent and backpropagation, learn the functionality in problem 1.

2.1 Challenges of the NAC_+ and $\text{NAC}_\bullet$

To simplify the problem we have chosen to leave out the gating mechanism and focus on the sub-units, assuming "oracle gating". The gating mechanism of the NALU is difficult to converge, as shown in table 2. This is likely due to the vastly different convergence properties. The independence in addition is a lot easier to learn than the heavily dependent multiplication operations, even for problems where addition is not the desired operation. Because addition converges faster, it becomes the best estimator early on and the gate will then choose addition. This causes the multiplication unit to be completely ignored, making it impossible for it to converge even if it is the desired operation.

2.1.1 Weight matrix construction

The weight matrix construction $\tanh(\hat{W}_{h_{\ell-1},h_\ell})\sigma(\hat{M}_{h_{\ell-1},h_\ell})$ has the following properties that could make convergence challenging using gradient decent. Firstly, the gradient of the loss function, $\mathcal{L}$, with respect to the weight matrices, $\hat{W}_{h_{\ell-1},h_\ell}$ and $\hat{M}_{h_{\ell-1},h_\ell}$, can be derived from equation 2 (derivation in appendix A.1).

$$\begin{aligned}
\frac{\partial \mathcal{L}}{\partial \hat{W}_{h_{\ell-1},h_\ell}} &= \frac{\partial \mathcal{L}}{\partial W_{h_{\ell-1},h_\ell}}(1 - \tanh^2(\hat{W}_{h_{\ell-1},h_\ell}))\sigma(\hat{M}_{h_{\ell-1},h_\ell}) \\
\frac{\partial \mathcal{L}}{\partial \hat{M}_{h_{\ell-1},h_\ell}} &= \frac{\partial \mathcal{L}}{\partial W_{h_{\ell-1},h_\ell}} \tanh(\hat{W}_{h_{\ell-1},h_\ell})\sigma(\hat{M}_{h_{\ell-1},h_\ell})(1 - \sigma(\hat{M}_{h_{\ell-1},h_\ell}))
\end{aligned} \tag{5}$$

To satisfy the desired property of $E[z_{h_\ell}] = 0$, which is important for convergence (**?**), then $\hat{W}_{h_{\ell-1},h_\ell}$ must be initialized to have zero expectation. However, for this initialization the gradient of the loss with respect to $\hat{M}_{h_{\ell-1},h_\ell}$ becomes zero.

Secondly, in our empirical analysis (see table 2) we find that equation 2 does not create the desired bias for $\{-1, 0, 1\}$ for the addition and subtraction problem.

To create the desired bias of $\{-1, 0, 1\}$ we add a biasing regularizer to the loss function, $\mathcal{L} = \hat{\mathcal{L}} + \lambda_{\text{sparse}}\mathcal{R}_{\ell,\text{sparse}}$. To prevent the gradient challenges when $E[\hat{W}_{h_{\ell-1},h_\ell}] = 0$ we propose a

simple linear construction, where $W_{h_{\ell-1},h_\ell}$ is clamped to $[-1, 1]$ in each iteration.

$$W_{h_{\ell-1},h_\ell} = \min(\max(W_{h_{\ell-1},h_\ell}, -1), 1), \tag{6}$$

$$\mathcal{R}_{\ell,\text{sparse}} = \frac{1}{H_\ell \cdot H_{\ell-1}} \sum_{h_\ell=1}^{H_\ell} \sum_{h_{\ell-1}=1}^{H_{\ell-1}} \min\left(|W_{h_{\ell-1},h_\ell}|, 1 - \left|W_{h_{\ell-1},h_\ell}\right|\right) \tag{7}$$

$$\text{NAU}: \ z_{h_\ell} = \sum_{h_{\ell-1}=1}^{H_{\ell-1}} W_{h_\ell,h_{\ell-1}} z_{h_{\ell-1}} \tag{8}$$

2.1.2 Challenges of division

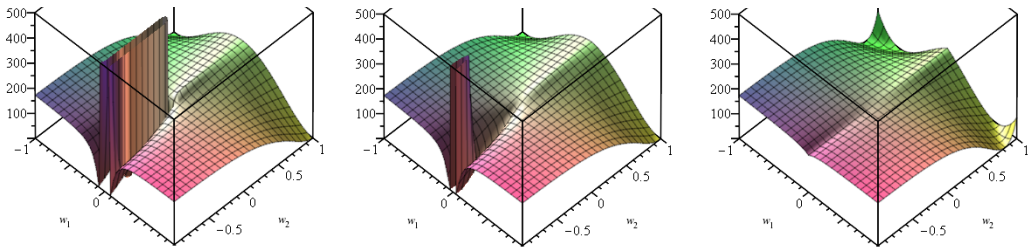
The $\text{NAC}_\bullet$, as formulated in equation 4, has the ability to perform exact division, or more precisely multiplication of the inverse of elements from a vector, when a weight in $W_{h_{\ell-1},h_\ell}$ is -1.

However, allowing this creates critical optimization challenges. Expanding the exp-log-transformation, $\text{NAC}_\bullet$ can be express as

$$\text{NAC}_\bullet: \ z_{h_\ell} = \prod_{h_{\ell-1}=1}^{H_{\ell-1}} \left(|z_{h_{\ell-1}}| + \epsilon\right)^{W_{h_\ell,h_{\ell-1}}} . \tag{9}$$

Equation (9) reveals that if $|z_{h_{\ell-1}}|$ is near zero (which is likely since $E[z_{h_{\ell-1}}] = 0$ is a desired property when initializing (**?**)), $W_{h_{\ell-1},h_\ell}$ is negative, and ϵ is small, then the output will explode. This issue is present even for a reasonably large ϵ value, and just a slightly negative $W_{h_{\ell-1},h_\ell}$, as visualized in figure 2. Also note that the curvature can cause convergence to an unstable area.

This singularity issue, is present even when multiplication is the desired operation. Since a multiplication unit should not explode for $z_{h_{\ell-1}}$ near zero, it is likely that supporting division is infeasible.



(a) $\text{NAC}_\bullet$ with $\epsilon = 10^{-7}$ (b) $\text{NAC}_\bullet$ with $\epsilon = 0.1$ (c) $\epsilon = 1$

Figure 2: RMS loss curvature for a NAC_+ layer followed by either a $\text{NAC}_\bullet$ or a NMU layer. The weight matrices constrained are to $\mathbf{W}_1 = \left[\begin{smallmatrix} w_1 & w_1 & 0 & 0 \\ w_1 & w_1 & w_1 & w_1 \end{smallmatrix}\right]$, $\mathbf{W}_2 = \left[\begin{smallmatrix} w_2 & w_2 \end{smallmatrix}\right]$. The problem is $x = (1, 1.2, 1.8, 2), t = 13.2$. The desired solution is $w_1 = w_2 = 1$, although this problem have additional undesired unstable solutions.

2.1.3 Initialization of $\text{NAC}_\bullet$

Initialization is important to consider for fast and consistent convergence, one desired property is that weights can be initialized such that $E[z_{h_\ell}] = 0$ (**?**). Using second order Taylor approximation and assuming all $z_{h_{\ell-1}}$ are uncorrelated, the expectation of $\text{NAC}_\bullet$ can be estimated as

$$E[z_{h_\ell}] \approx \left(1 + \frac{1}{2} Var[W_{h_\ell,h_{\ell-1}}] \log(|E[z_{h_{\ell-1}}]| + \epsilon)^2\right)^{H_{\ell-1}} \Rightarrow E[z_{h_\ell}] > 1. \tag{10}$$

As shown in equation 10, satisfying $E[z_{h_\ell}] = 0$ for $\text{NAC}_\bullet$ is likely impossible. The variance can also not be input-independent initialized and is expected to explode (proofs in Appendix B.3).

2.1.4 NEURAL MULTIPLICATION UNIT

To solve the the gradient and initialization challenges for $\mathrm{NAC}_\bullet$ we propose a new neural multiplication unit (NMU):

$$W_{h_{\ell-1},h_\ell} = \min(\max(W_{h_{\ell-1},h_\ell}, 0), 1), \tag{11}$$

$$\mathcal{R}_{\ell,\text{sparse}} = \frac{1}{H_\ell \cdot H_{\ell-1}} \sum_{h_\ell=1}^{H_\ell} \sum_{h_{\ell-1}=1}^{H_{\ell-1}} \min\left(W_{h_{\ell-1},h_\ell}, 1 - W_{h_{\ell-1},h_\ell}\right) \tag{12}$$

$$\text{NMU}: \ z_{h_\ell} = \prod_{h_{\ell-1}=1}^{H_{\ell-1}} \left(W_{h_{\ell-1},h_\ell} z_{h_{\ell-1}} + 1 - W_{h_{\ell-1},h_\ell}\right) \tag{13}$$

The NMU is regularized similar to the NAU and has a multiplicative identity when $W_{h_{\ell-1},h_\ell} = 0$. The NMU unit does not support division by design. As opposed to the $\mathrm{NAC}_\bullet$, the NMU can represent input of both negative and positive values and is not ϵ bounded, which allows the NMU to extrapolate to $z_{h_{\ell-1}}$ that are negative or smaller than ϵ. Its gradients are derived in Appendix A.3.

2.1.5 MOMENTS AND INITIALIZATION

Our proposed NAU can be initialized using Glorot initialization as it is a linear layer. The NAC_+ unit can also achieve an ideal initialization, although it is less trivial (details in Appendix B.2).

Our proposed NMU is initialized with $E[W_{h_\ell,h_{\ell-1}}] = 1/2$. Assuming all $z_{h_{\ell-1}}$ are uncorrelated, and $E[z_{h_{\ell-1}}] = 0$, which is the case for most units, the expectation can be approximated to

$$E[z_{h_\ell}] \approx \left(\frac{1}{2}\right)^{H_{\ell-1}}, \tag{14}$$

which approaches zero for $H_{\ell-1} \to \infty$ (details in Appendix B.4). The NMU can, with the assumption that, $Var[z_{h_{\ell-1}}] = 1$ and $H_{\ell-1}$ is large, be initialized optimally with $Var[W_{h_{\ell-1},h_\ell}] = \frac{1}{4}$ (proof in Appendix B.4.3).

2.1.6 REGULARIZER SCALING

We use the regularizer scaling as defined in (15). The motivation here, is that the optimization consists of two parts. A warmup period, where $W_{h_{\ell-1},h_\ell}$ should get close to the solution, unhindered by the sparsity regularizer, and then a period where the solution is made sparse.

$$\lambda_{\text{sparse}} = \hat{\lambda}_{\text{sparse}} \max(\min(\frac{t - \lambda_{\text{start}}}{\lambda_{\text{end}} - \lambda_{\text{start}}}, 1), 0) \tag{15}$$

3 RELATED WORK

Pure neural models using convolutions, gating, differentiable memory, and/or attention architectures have attempted to learn arithmetic tasks through backpropagation (**???**). Some of the results have close to perfect extrapolation. However, the models are constrained to work only on whole numbers and requires well defined arithmetic setups such as binary representations of numbers for input and output. We do not extend current approximative methods, but instead develop two new units that can learn exact arithmetic operations on real numbers, without requiring a binary representations.

The Neural Arithmetic Expression Calculator (**?**) propose learning real number arithmetic by having neural network subcomponents and repeatedly combine them through a memory-encoder-decoder architecture learned with hierarchical reinforcement learning. While this model has the ability to dynamically handle a larger variety of expressions compared to our solution their solution do not generalize much beyond interpolation length.

In our experiments, the NAU is used to do a subset-selection, which is then followed by either a summation or multiplication. An alternative, fully differentiable version, is to use a gumbel-softmax that can perform exact subset-selection (**?**). However, this is restricted to a predefined subset size, which is a strong assumption that our units are not limited by.

4 EXPERIMENTAL RESULTS

4.1 ARITHMETIC DATASETS

The arithmetic dataset is a replica of the "simple function task" as shown in (**?**). The goal is to sum two random subsets of a vector and perform a arithmetic operation as defined below

$$t = \sum_{i=s_{1,\text{start}}}^{s_{1,\text{end}}} x_i \circ \sum_{i=s_{2,\text{start}}}^{s_{2,\text{end}}} x_i \quad \text{where } \mathbf{x} \in \mathbb{R}^n, x_i \sim \text{Uniform}[r_{\text{lower}}, r_{\text{upper}}], \circ \in \{+, -, \times\} \tag{16}$$

where n (default 100), $U[r_{\text{lower}}, r_{\text{upper}}]$ (interpolation default is $U[1, 2]$ and extrapolation default is $U[2, 6]$), $\circ$, the subset size (default 25%), and subset overlap (default 50%) are dataset parameters that we use to assess learning capability (see details in appendix C.1 and the effect of these parameters in appendix C.3).

4.1.1 MODEL EVALUATION

The goal is to achieve a solution that is acceptably close to a perfect solution. To evaluate if a model instance solves the task consistently we compare the MSE to a nearly-perfect solution on the extrapolation range over multiple seeds. If $\mathbf{W}_1, \mathbf{W}_2$ defines the weights of the fitted model, and $\mathbf{W}_1^\epsilon$ is nearly-perfect and $\mathbf{W}_2^*$ is perfect (example in equation 17), the success criteria is $\mathcal{L}_{\mathbf{W}_1,\mathbf{W}_2} < \mathcal{L}_{\mathbf{W}_1^\epsilon,\mathbf{W}_2^*}$, measured on the extrapolation error, for $\epsilon = 10^{-5}$.

$$\mathbf{W}_1^\epsilon = \begin{bmatrix} 1-\epsilon & 1-\epsilon & 0+\epsilon & 0+\epsilon \\ 1-\epsilon & 1-\epsilon & 1-\epsilon & 1-\epsilon \end{bmatrix}, \mathbf{W}_2^* = [1 \quad 1] \tag{17}$$

To measure speed of convergence the first iteration for which $\mathcal{L}_{\mathbf{W}_1,\mathbf{W}_2} < \mathcal{L}_{\mathbf{W}_1^\epsilon,\mathbf{W}_2^*}$ is reported with a 95% confidence interval. Only models that managed to solve the task are considered.

We assume an approximate discrete solution with parameters close to $\{-1, 0, 1\}$ is important for inferring exact arithmetic operations. To measure the sparsity we introduce a sparsity error (defined in equation 18). Similar to the convergence metric we only considered model instances that did solve the task and report the 95% confidence interval.

$$E_{\text{sparsity}} = \max_{h_{\ell-1},h_\ell} \min(|W_{h_{\ell-1},h_\ell}|, |1 - |W_{h_{\ell-1},h_\ell}||) \tag{18}$$

We evaluate each metric every 1000 iterations based on the test set that uses the extrapolation range, and choose the best iteration based on the validation dataset that uses the interpolation range.

4.1.2 ARITHMETIC OPERATION COMPARISON

We compare models on different arithmetic operation $\circ \in \{+, -, \times\}$. The multiplication models, NMU and $\text{NAC}_\bullet$, have an addition layer first, either NAU or NAC_+, followed by a multiplication layer. The addition models are just two layers of the same unit. Finally, the NALU model consists of two NALU layers. See explicit definitions in appendix C.2.

Each experiment is trained for $5 \cdot 10^6$ iterations. More experimental details can be found in appendix C.2. Results are presented in table 2. Comparison with more models can be found in appendix C.2 and an ablation study of the NMU can be found in appendix C.4.

For multiplication, the NMU succeeds more often and converges faster. For addition and subtraction, the NAU model converges faster, given the median, and has a sparser solution.

4.1.3 EVALUATING THEORETICAL CLAIMS

To validate our theoretical claim, that the NMU models works better than $NAC_\bullet$ for larger $H_{\ell-1}$, we increase the hidden size of the network, thereby adding redundant units. Redundant units are very common neural networks, where the purpose is to fit an unknown function.

Additionally, the NMU model is unlike the $NAC_\bullet$ model also capable of understanding inputs that are both negative and positive. To validate this empirically, the training and validation datasets are sampled for $\text{U}[-2, 2]$, and then tested on $\text{U}[-6, -2] \cup \text{U}[2, 6]$.

Table 2: Shows the success-rate for $\mathcal{L}_{\mathbf{W}_1,\mathbf{W}_2} < \mathcal{L}_{\mathbf{W}_1^\epsilon,\mathbf{W}_2^*}$, at what global step the model converged at and the sparsity error for all weight matrices, with 95% confidence interval. Best result is highlighted without considering significance.

Op	Model	Success	Solved at		Sparsity error
		Rate	Median	Mean	Mean
$\times$	$\mathrm{NAC}_\bullet$	24%	$2.9 \cdot 10^6$	$3.0 \cdot 10^6 \pm 6.6 \cdot 10^5$	$4.0 \cdot 10^{-4} \pm 4.1 \cdot 10^{-4}$
	NALU	0%	—	—	—
	NMU	**100%**	$\mathbf{1.5 \cdot 10^6}$	$\mathbf{1.7 \cdot 10^6 \pm 3.1 \cdot 10^5}$	$\mathbf{4.4 \cdot 10^{-7} \pm 6.9 \cdot 10^{-8}}$
$+$	NAC_+	**100%**	$2.4 \cdot 10^5$	$2.6 \cdot 10^5 \pm 4.9 \cdot 10^4$	$4.7 \cdot 10^{-1} \pm 2.4 \cdot 10^{-2}$
	Linear	**100%**	$5.8 \cdot 10^4$	$\mathbf{6.0 \cdot 10^4 \pm 4.2 \cdot 10^3}$	$5.5 \cdot 10^{-1} \pm 4.1 \cdot 10^{-2}$
	NALU	4%	$1.8 \cdot 10^6$	$1.8 \cdot 10^6$	$4.9 \cdot 10^{-1}$
	NAU	**100%**	$\mathbf{1.8 \cdot 10^4}$	$1.8 \cdot 10^5 \pm 2.4 \cdot 10^5$	$\mathbf{3.2 \cdot 10^{-5} \pm 2.8 \cdot 10^{-5}}$
$-$	NAC_+	**100%**	$9.0 \cdot 10^3$	$6.9 \cdot 10^5 \pm 5.2 \cdot 10^5$	$4.5 \cdot 10^{-1} \pm 2.5 \cdot 10^{-2}$
	Linear	4%	$2.4 \cdot 10^6$	$2.4 \cdot 10^6$	$4.8 \cdot 10^{-1}$
	NALU	8%	$2.3 \cdot 10^6$	$2.3 \cdot 10^6 \pm 1.1 \cdot 10^7$	$4.7 \cdot 10^{-1} \pm 3.2 \cdot 10^{-1}$
	NAU	**100%**	$\mathbf{5.0 \cdot 10^3}$	$\mathbf{1.8 \cdot 10^5 \pm 3.6 \cdot 10^5}$	$\mathbf{1.8 \cdot 10^{-1} \pm 1.0 \cdot 10^{-1}}$

Finally, to validate that division and the lack of bias in $NAC_\bullet$ are critical issues but that solving these alone are not enough, two additional models are added to the comparison. A variant called $\mathrm{NAC}_{\bullet,\sigma}$ that only supports multiplication, by just constraining the weights with $W = \sigma(\hat{W})$ in $\mathrm{NAC}_\bullet$. And a variant of $\mathrm{NAC}_{\bullet,\sigma}$ that uses linear weights and bias regularization, identically to that in NMU model, this model is called $\mathrm{NAC}_{\bullet,\mathrm{NMU}}$.

Figure 3 shows that the NMU can both handle a much larger hidden-size, as well as negative inputs, and that solving the division and bias issues alone improves the success rate, but are no enough.

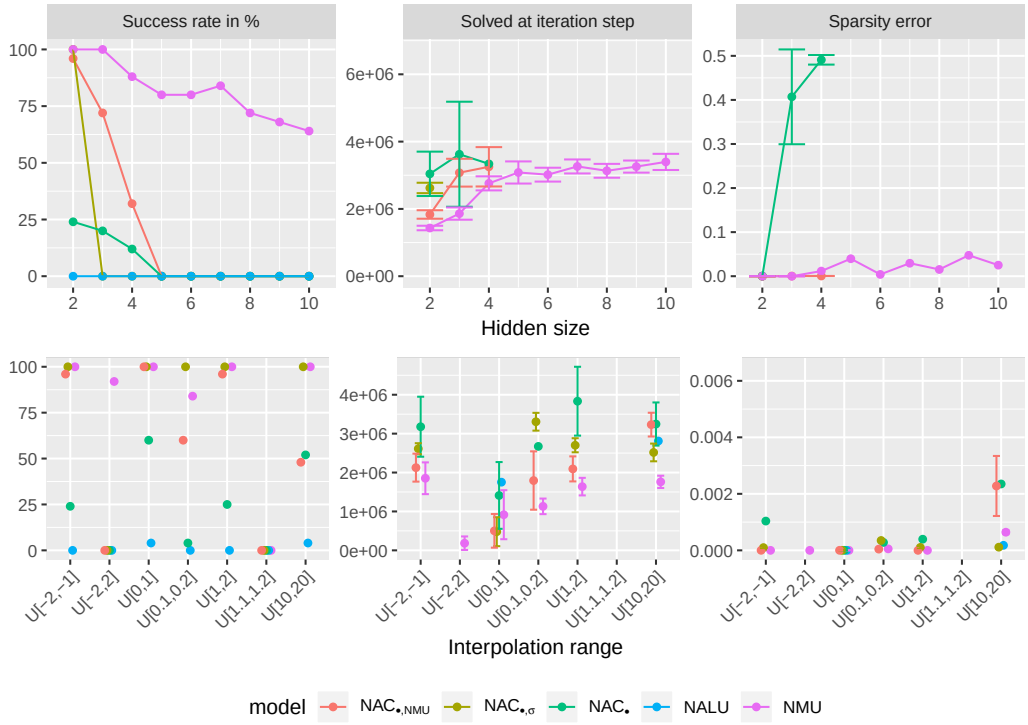


Figure 3: Shows that the NMU can handle a large hidden size, and works when the input contains both positive and negative numbers ($U[-2,-2]$).

4.2 PRODUCT OF SEQUENTIAL MNIST

To evaluate the NMU's ability to backpropergate though a larger unconstrained neural network, the NMU is applied as a recurrent-unit to a sequence of MNIST digits, where the target is to fit the cumulative product. This task is similar to "MNIST Counting and Arithmetic Tasks" in (**?**), but use multiplication rather than addition, the same CNN model is also used [2]. Each model is trained on sequences of length 2, and then tested on sequences of length 20.

Success of convergence is determined by comparing the MSE of each model, with a baseline model that directly computes the sequential product. If the MSE of each model, is less than the upper 1% MSE-confidence-interval of the baseline model, then the model is considered successfully converged.

Sparsity and "solved at iteration step" is determined as described in experiment 4.1. The validation set, is the last 5000 MNIST digits from the training set, which is used to select the best epoch.

In this experiment we discovered that having an unconstrained "input-network" causes the multiplication-units to learn an undesired solution, such as $(0.1 \cdot 81 + 1 - 0.1) = 9$. This solves the problem with a similar success-rate, but not in the intended way. To prevent such solution, we regularize the CNN output with $\mathcal{R}_z = \frac{1}{H_{\ell-1}H_\ell} \sum_{h_\ell}^{H_\ell} \sum_{h_{\ell-1}}^{H_{\ell-1}} (1 - W_{h_{\ell-1},h_\ell}) \cdot (1 - \bar{z}_{h_{\ell-1}})^2$. This regularizer is applied to the NMU and $\mathrm{NAC}_{\bullet,\mathrm{NMU}}$ models. See appendix D.1 for the results, when this regularizer is not used.

Figure 4 shows that the NMU does not hindre learning a more complex neural network. And that it can extrapolate to much longer sequences than what is is trained on.

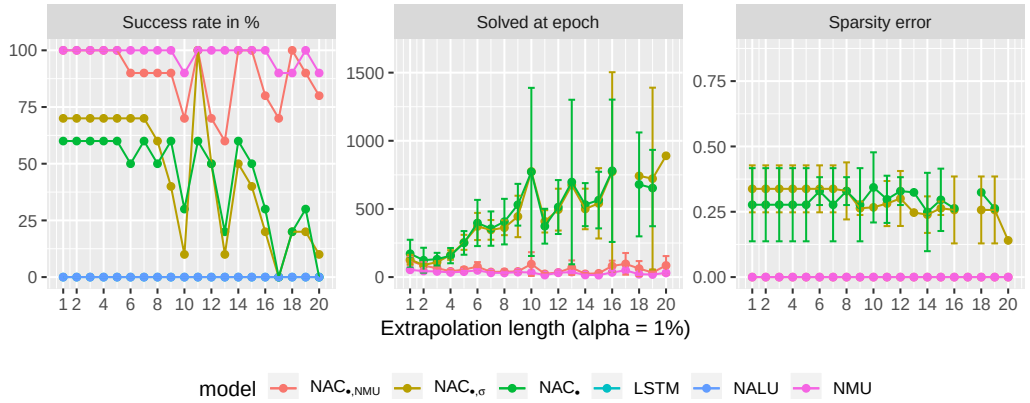


Figure 4: Shows the ability of each model to backpropergation and extrapolate to larger sequence lengths.

5 CONCLUSION

We have investigated how a function based on constraint weights can extrapolate well on arithmetic operations and if it can learn the weights by stochastic gradient descent. Inspired by the recently proposed Neural Arithmetic Logic Unit (NALU), we propose the Neural Addition Unit (NAU) and Neural Multiplication Unit (NMU), which has more consistent convergence on a range of addition, subtraction, and multiplication tasks than the NALU. Beyond convergence capabilities, our modifications makes the NMU capable of extrapolating to numerically small numbers and the negative range, which has previously been impossible. Our analytic analysis highlights that learning division is challenging as division close to zero creates explosions.

[2] https://github.com/pytorch/examples/tree/master/mnist

ACKNOWLEDGMENTS

We would like to thank Andrew Trask and the other authors of the NALU paper, for highlighting the importance and challenges of extrapolation in Neural Networks. We would also like to thank the students Raja Shan Zaker Kreen and William Frisch Møller from The Technical University of Denmark, who initially showed us that the NALU does not converge consistently.

A GRADIENT DERIVATIVES

A.1 WEIGHT MATRIX CONSTRUCTION

For clarity the weight matrix construction is defined using scalar notation

$$W_{h_\ell,h_{\ell-1}} = \tanh(\hat{W}_{h_\ell,h_{\ell-1}})\sigma(\hat{M}_{h_\ell,h_{\ell-1}}) \tag{19}$$

The of the loss with respect to $\hat{W}_{h_\ell,h_{\ell-1}}$ and $\hat{M}_{h_\ell,h_{\ell-1}}$ is then straight forward to derive.

$$\begin{aligned}
\frac{\partial \mathcal{L}}{\partial \hat{W}_{h_\ell,h_{\ell-1}}} &= \frac{\partial \mathcal{L}}{\partial W_{h_\ell,h_{\ell-1}}}\frac{\partial W_{h_\ell,h_{\ell-1}}}{\partial \hat{W}_{h_\ell,h_{\ell-1}}} \\
&= \frac{\partial \mathcal{L}}{\partial W_{h_\ell,h_{\ell-1}}}(1-\tanh^2(\hat{W}_{h_\ell,h_{\ell-1}}))\sigma(\hat{M}_{h_\ell,h_{\ell-1}}) \\
\frac{\partial \mathcal{L}}{\partial \hat{M}_{h_\ell,h_{\ell-1}}} &= \frac{\partial \mathcal{L}}{\partial W_{h_\ell,h_{\ell-1}}}\frac{\partial W_{h_\ell,h_{\ell-1}}}{\partial \hat{M}_{h_\ell,h_{\ell-1}}} \\
&= \frac{\partial \mathcal{L}}{\partial W_{h_\ell,h_{\ell-1}}}\tanh(\hat{W}_{h_\ell,h_{\ell-1}})\sigma(\hat{M}_{h_\ell,h_{\ell-1}})(1-\sigma(\hat{M}_{h_\ell,h_{\ell-1}}))
\end{aligned} \tag{20}$$

As seen from this result, one only needs to consider $\frac{\partial \mathcal{L}}{\partial W_{h_\ell,h_{\ell-1}}}$ for NAC_+ and $\mathrm{NAC}_\bullet$, as the gradient with respect to $\hat{W}_{h_\ell,h_{\ell-1}}$ and $\hat{M}_{h_\ell,h_{\ell-1}}$ is just a multiplication on $\frac{\partial \mathcal{L}}{\partial W_{h_\ell,h_{\ell-1}}}$.

A.2 GRADIENT OF $\mathrm{NAC}_\bullet$

First the $\mathrm{NAC}_\bullet$ is defined using scalar notation.

$$z_{h_\ell} = \exp\left(\sum_{h_{\ell-1}=1}^{H_{\ell-1}} W_{h_\ell,h_{\ell-1}}\log(|z_{h_{\ell-1}}|+\epsilon)\right) \tag{21}$$

The gradient of the loss with respect to $W_{h_\ell,h_{\ell-1}}$ is straight forward to derive.

$$\begin{aligned}
\frac{\partial z_{h_\ell}}{\partial W_{h_\ell,h_{\ell-1}}} &= \exp\left(\sum_{h'_{\ell-1}=1}^{H_{\ell-1}} W_{h_\ell,h'_{\ell-1}}\log(|z_{h'_{\ell-1}}|+\epsilon)\right)\log(|z_{h_{\ell-1}}|+\epsilon) \\
&= z_{h_\ell}\log(|z_{h_{\ell-1}}|+\epsilon)
\end{aligned} \tag{22}$$

We now wish to derive the backpropagation term $\delta_{h_\ell} = \frac{\partial \mathcal{L}}{\partial z_{h_\ell}}$, because z_{h_ℓ} affects $\{z_{h_{\ell+1}}\}_{h_{\ell+1}=1}^{H_{\ell+1}}$ this becomes:

$$\delta_{h_\ell} = \frac{\partial \mathcal{L}}{\partial z_{h_\ell}} = \sum_{h_{\ell+1}=1}^{H_{\ell+1}}\frac{\partial \mathcal{L}}{\partial z_{h_{\ell+1}}}\frac{\partial z_{h_{\ell+1}}}{\partial z_{h_\ell}} = \sum_{h_{\ell+1}=1}^{H_{\ell+1}}\delta_{h_{\ell+1}}\frac{\partial z_{h_{\ell+1}}}{\partial z_{h_\ell}} \tag{23}$$

To make it easier to derive $\frac{\partial z_{h_{\ell+1}}}{\partial z_{h_\ell}}$ we re-express the z_{h_ℓ} as $z_{h_{\ell+1}}$.

$$z_{h_{\ell+1}} = \exp\left(\sum_{h_\ell=1}^{H_\ell} W_{h_{\ell+1},h_\ell}\log(|z_{h_\ell}|+\epsilon)\right) \tag{24}$$

The gradient of $\frac{\partial z_{h_{\ell+1}}}{\partial z_{h_\ell}}$ is then:

$$\begin{aligned}
\frac{\partial z_{h_{\ell+1}}}{\partial z_{h_\ell}} &= \exp\left(\sum_{h_\ell=1}^{H_\ell} W_{h_{\ell+1},h_\ell} \log(|z_{h_\ell}| + \epsilon)\right) W_{h_{\ell+1},h_\ell} \frac{\partial \log(|z_{h_\ell}| + \epsilon)}{\partial z_{h_\ell}} \\
&= \exp\left(\sum_{h_\ell=1}^{H_\ell} W_{h_{\ell+1},h_\ell} \log(|z_{h_\ell}| + \epsilon)\right) W_{h_{\ell+1},h_\ell} \frac{\mathrm{abs}'(z_{h_\ell})}{|z_{h_\ell}| + \epsilon} \\
&= m_{h_{\ell+1}} W_{h_{\ell+1},h_\ell} \frac{\mathrm{abs}'(z_{h_\ell})}{|z_{h_\ell}| + \epsilon}
\end{aligned} \tag{25}$$

$\mathrm{abs}'(z_{h_\ell})$ is the gradient of the absolute function. In the paper we denote this as $\mathrm{sign}(z_{h_\ell})$ for brevity. However, depending on the exact definition used there may be a difference for $z_{h_\ell} = 0$, as $\mathrm{abs}'(0)$ is undefined. In practicality this doesn't matter much though, although theoretically it does mean that the expectation of this is theoretically undefined when $E[z_{h_\ell}] = 0$.

A.3 Gradient of NMU

In scalar notation the NMU is defined as:

$$z_{h_\ell} = \prod_{h_{\ell-1}=1}^{H_{\ell-1}} \left(W_{h_{\ell-1},h_\ell} z_{h_{\ell-1}} + 1 - W_{h_{\ell-1},h_\ell}\right) \tag{26}$$

The gradient of the loss with respect to $W_{h_{\ell-1},h_\ell}$ is fairly trivial. Note that every term but the one for $h_{\ell-1}$, is just a constant with respect to $W_{h_{\ell-1},h_\ell}$. The product, expect the term for $h_{\ell-1}$ can be expressed as $\frac{z_{h_\ell}}{W_{h_{\ell-1},h_\ell} z_{h_{\ell-1}} + 1 - W_{h_{\ell-1},h_\ell}}$. Using this fact, it becomes trivial to derive the gradient as:

$$\frac{\partial \mathcal{L}}{\partial w_{h_\ell,h_{\ell-1}}} = \frac{\partial \mathcal{L}}{\partial z_{h_\ell}} \frac{\partial z_{h_\ell}}{\partial w_{h_\ell,h_{\ell-1}}} = \frac{\partial \mathcal{L}}{\partial z_{h_\ell}} \frac{z_{h_\ell}}{W_{h_{\ell-1},h_\ell} z_{h_{\ell-1}} + 1 - W_{h_{\ell-1},h_\ell}} \left(z_{h_{\ell-1}} - 1\right) \tag{27}$$

Similarly, the gradient $\frac{\partial \mathcal{L}}{\partial z_{h_\ell}}$ which is essential in backpropagation can equally easily be derived as:

$$\frac{\partial \mathcal{L}}{\partial z_{h_{\ell-1}}} = \sum_{h_\ell=1}^{H_\ell} \frac{\partial \mathcal{L}}{\partial z_{h_\ell}} \frac{\partial z_{h_\ell}}{\partial z_{h_{\ell-1}}} = \sum_{h_\ell=1}^{H_\ell} \frac{z_{h_\ell}}{W_{h_{\ell-1},h_\ell} z_{h_{\ell-1}} + 1 - W_{h_{\ell-1},h_\ell}} W_{h_{\ell-1},h_\ell} \tag{28}$$

B MOMENTS

B.1 OVERVIEW

B.1.1 MOMENTS AND INITIALIZATION FOR ADDITION

The desired properties for initialization are according to Glorot et al. (**?**):

$$\begin{aligned} E[z_{h_\ell}] &= 0 & E\left[\frac{\partial \mathcal{L}}{\partial z_{h_{\ell-1}}}\right] &= 0 \\ Var[z_{h_\ell}] &= Var\left[z_{h_{\ell-1}}\right] & Var\left[\frac{\partial \mathcal{L}}{\partial z_{h_{\ell-1}}}\right] &= Var\left[\frac{\partial \mathcal{L}}{\partial z_{h_\ell}}\right] \end{aligned} \tag{29}$$

B.1.2 INITIALIZATION FOR ADDITION

Glorot initialization can not be used for NAC_+ as $W_{h_{\ell-1},h_\ell}$ is not sampled directly. Assuming that $\hat{W}_{h_\ell,h_{\ell-1}} \sim \mathrm{Uniform}[-r, r]$ and $\hat{M}_{h_\ell,h_{\ell-1}} \sim \mathrm{Uniform}[-r, r]$, then the variance can be derived (see proof in Appendix B.2) to be:

$$Var[W_{h_{\ell-1},h_\ell}] = \frac{1}{2r}\left(1 - \frac{\tanh(r)}{r}\right)\left(r - \tanh\left(\frac{r}{2}\right)\right) \tag{30}$$

One can then solve for r, given the desired variance ($Var[W_{h_{\ell-1},h_\ell}] = \frac{2}{H_{\ell-1}+H_\ell}$) (**?**).

B.1.3 MOMENTS AND INITIALIZATION FOR MULTIPLICATION

Using second order multivariate Taylor approximation and some assumptions of uncorrelated stochastic variables, the expectation and variance of the $\mathrm{NAC}_\bullet$ layer can be estimated to:

$$\begin{aligned} f(c_1, c_2) &= \left(1 + c_1 \frac{1}{2} Var[W_{h_\ell,h_{\ell-1}}] \log(|E[z_{h_{\ell-1}}]| + \epsilon)^2\right)^{c_2\ H_{\ell-1}} \\ E[z_{h_\ell}] &\approx f\left(1, 1\right) \\ Var[z_{h_\ell}] &\approx f\left(4, 1\right) - f\left(1, 2\right) \\ E\left[\frac{\partial \mathcal{L}}{\partial z_{h_{\ell-1}}}\right] &= 0 \\ Var\left[\frac{\partial \mathcal{L}}{\partial z_{h_{\ell-1}}}\right] &\approx Var\left[\frac{\partial \mathcal{L}}{\partial z_{h_\ell}}\right] H_\ell\ f\left(4, 1\right)\ Var[W_{h_\ell,h_{\ell-1}}] \\ &\quad \cdot \left(\frac{1}{\left(|E[z_{h_{\ell-1}}]| + \epsilon\right)^2} + \frac{3}{\left(|E[z_{h_{\ell-1}}]| + \epsilon\right)^4} Var[z_{h_{\ell-1}}]\right) \end{aligned} \tag{31}$$

This is problematic because $E[z_{h_\ell}] \geq 1$, and the variance explodes for $E[z_{h_{\ell-1}}] = 0$. $E[z_{h_{\ell-1}}] = 0$ is normally a desired property (**?**). The variance explodes for $E[z_{h_{\ell-1}}] = 0$, and can thus not be initialized to anything meaningful.

For our proposed NMU, the expectation and variance can be derived (see proof in Appendix B.4) using the same assumptions as before, although no Taylor approximation is required:

$$\begin{aligned}
E[z_{h_\ell}] &\approx \left(\frac{1}{2}\right)^{H_{\ell-1}} \\
E\left[\frac{\partial \mathcal{L}}{\partial z_{h_{\ell-1}}}\right] &\approx 0 \\
Var[z_{h_\ell}] &\approx \left(Var[W_{h_{\ell-1},h_\ell}] + \frac{1}{4}\right)^{H_{\ell-1}} \left(Var[z_{h_{\ell-1}}] + 1\right)^{H_{\ell-1}} - \left(\frac{1}{4}\right)^{H_{\ell-1}} \\
Var\left[\frac{\partial \mathcal{L}}{\partial z_{h_{\ell-1}}}\right] &\approx Var\left[\frac{\partial \mathcal{L}}{\partial z_{h_\ell}}\right] H_\ell \\
&\quad \cdot \left(\left(Var[W_{h_{\ell-1},h_\ell}] + \frac{1}{4}\right)^{H_{\ell-1}} \left(Var[z_{h_{\ell-1}}] + 1\right)^{H_{\ell-1}-1}\right)
\end{aligned} \tag{32}$$

These expectations are better behaved. It is properly unlikely to expect that the expectation can become zero, since the identity for multiplication is 1. However, for a large $H_{\ell-1}$ it will be near zero.

The variance is also better behaved, but does not provide a input-independent initialization strategy. We propose initializing with $Var[W_{h_{\ell-1},h_\ell}] = \frac{1}{4}$, as this is the solution to $Var[z_{h_\ell}] = Var[z_{h_{\ell-1}}]$ assuming $Var[z_{h_{\ell-1}}] = 1$ and a large $H_{\ell-1}$ (see proof in Appendix B.4.3). However, feel free to compute more exact solutions.

B.2 Expectation and variance for weight matrix construction in NAC layers

The weight matrix construction in NAC, is defined in scalar notation as:

$$W_{h_\ell,h_{\ell-1}} = \tanh(\hat{W}_{h_\ell,h_{\ell-1}})\sigma(\hat{M}_{h_\ell,h_{\ell-1}}) \tag{33}$$

Simplifying the notation of this, and re-expressing it using stochastic variables with uniform distributions this can be written as:

$$\begin{aligned}
W &\sim \tanh(\hat{W})\sigma(\hat{M}) \\
\hat{W} &\sim U[-r, r] \\
\hat{M} &\sim U[-r, r]
\end{aligned} \tag{34}$$

Since $\tanh(\hat{W})$ is an odd-function and $E[\hat{W}] = 0$, deriving the expectation $E[W]$ is trivial.

$$\mathrm{E}[W] = \mathrm{E}[\tanh(\hat{W})]\mathrm{E}[\sigma(\hat{M})] = 0 \cdot \mathrm{E}[\sigma(\hat{M})] = 0 \tag{35}$$

The variance is more complicated, however as $\hat{W}$ and $\hat{M}$ are independent, it can be simplified to:

$$\mathrm{Var}[W] = \mathrm{E}[\tanh(\hat{W})^2]\mathrm{E}[\sigma(\hat{M})^2] - \mathrm{E}[\tanh(\hat{W})]^2\mathrm{E}[\sigma(\hat{M})]^2 = \mathrm{E}[\tanh(\hat{W})^2]\mathrm{E}[\sigma(\hat{M})^2] \tag{36}$$

These second moments can be analyzed independently. First for $\mathrm{E}[\tanh(\hat{W})^2]$:

$$\begin{aligned}
\mathrm{E}[\tanh(\hat{W})^2] &= \int_{-\infty}^{\infty} \tanh(x)^2 f_{U[-r,r]}(x)\,\mathrm{d}x \\
&= \frac{1}{2r}\int_{-r}^{r} \tanh(x)^2\,\mathrm{d}x \\
&= \frac{1}{2r} \cdot 2 \cdot (r - \tanh(r)) \\
&= 1 - \frac{\tanh(r)}{r}
\end{aligned} \tag{37}$$

Then for $\mathrm{E}[\tanh(\hat{M})^2]$:

$$\begin{aligned}\mathrm{E}[\sigma(\hat{M})^2] &= \int_{-\infty}^{\infty} \sigma(x)^2 f_{U[-r,r]}(x)\,\mathrm{d}x \\ &= \frac{1}{2r}\int_{-r}^{r} \sigma(x)^2\,\mathrm{d}x \\ &= \frac{1}{2r}\left(r - \tanh\left(\frac{r}{2}\right)\right)\end{aligned} \tag{38}$$

Finally this gives the variance:

$$\mathrm{Var}[W] = \frac{1}{2r}\left(1 - \frac{\tanh(r)}{r}\right)\left(r - \tanh\left(\frac{r}{2}\right)\right) \tag{39}$$

B.3 Expectation and variance of NAC$_\bullet$

B.3.1 Forward pass

Expectation Assuming that each $z_{h_{\ell-1}}$ are uncorrelated the expectation can be simplified to:

$$\begin{aligned}E[z_{h_\ell}] &= E\left[\exp\left(\sum_{h_{\ell-1}=1}^{H_{\ell-1}} W_{h_\ell,h_{\ell-1}} \log(|z_{h_{\ell-1}}| + \epsilon)\right)\right] \\ &= E\left[\prod_{h_{\ell-1}=1}^{H_{\ell-1}} \exp(W_{h_\ell,h_{\ell-1}} \log(|z_{h_{\ell-1}}| + \epsilon))\right] \\ &\approx \prod_{h_{\ell-1}=1}^{H_{\ell-1}} E[\exp(W_{h_\ell,h_{\ell-1}} \log(|z_{h_{\ell-1}}| + \epsilon))] \\ &= E[\exp(W_{h_\ell,h_{\ell-1}} \log(|z_{h_{\ell-1}}| + \epsilon))]^{H_{\ell-1}} \\ &= E\left[(|z_{h_{\ell-1}}| + \epsilon)^{W_{h_\ell,h_{\ell-1}}}\right]^{H_{\ell-1}} \\ &= E\left[f(z_{h_{\ell-1}}, W_{h_\ell,h_{\ell-1}})\right]^{H_{\ell-1}}\end{aligned} \tag{40}$$

Here we define g as a non-linear transformation function of two independent stochastic variables:

$$f(z_{h_{\ell-1}}, W_{h_\ell,h_{\ell-1}}) = (|z_{h_{\ell-1}}| + \epsilon)^{W_{h_\ell,h_{\ell-1}}} \tag{41}$$

We then take the second order Taylor approximation of g, around $(E[z_{h_{\ell-1}}], E[W_{h_\ell,h_{\ell-1}}])$.

$$\begin{aligned}&E[f(z_{h_{\ell-1}}, W_{h_\ell,h_{\ell-1}})] \approx E\Bigg[\\ &g(E[z_{h_{\ell-1}}], E[W_{h_\ell,h_{\ell-1}}]) \\ &+ \begin{bmatrix} z_{h_{\ell-1}} - E[z_{h_{\ell-1}}] \\ W_{h_\ell,h_{\ell-1}} - E[W_{h_\ell,h_{\ell-1}}] \end{bmatrix}^T \begin{bmatrix} \frac{\partial g(z_{h_{\ell-1}}, W_{h_\ell,h_{\ell-1}})}{\partial z_{h_{\ell-1}}} \\ \frac{\partial g(z_{h_{\ell-1}}, W_{h_\ell,h_{\ell-1}})}{\partial W_{h_\ell,h_{\ell-1}}} \end{bmatrix} \Bigg|_{\substack{z_{h_{\ell-1}} = E[z_{h_{\ell-1}}] \\ W_{h_\ell,h_{\ell-1}} = E[W_{h_\ell,h_{\ell-1}}]}} \\ &+ \frac{1}{2}\begin{bmatrix} z_{h_{\ell-1}} - E[z_{h_{\ell-1}}] \\ W_{h_\ell,h_{\ell-1}} - E[W_{h_\ell,h_{\ell-1}}] \end{bmatrix}^T \\ &\bullet \begin{bmatrix} \frac{\partial^2 g(z_{h_{\ell-1}}, W_{h_\ell,h_{\ell-1}})}{\partial^2 z_{h_{\ell-1}}} & \frac{\partial^2 g(z_{h_{\ell-1}}, W_{h_\ell,h_{\ell-1}})}{\partial z_{h_{\ell-1}} \partial W_{h_\ell,h_{\ell-1}}} \\ \frac{\partial^2 g(z_{h_{\ell-1}}, W_{h_\ell,h_{\ell-1}})}{\partial z_{h_{\ell-1}} \partial W_{h_\ell,h_{\ell-1}}} & \frac{\partial^2 g(z_{h_{\ell-1}}, W_{h_\ell,h_{\ell-1}})}{\partial^2 W_{h_\ell,h_{\ell-1}}} \end{bmatrix} \Bigg|_{\substack{z_{h_{\ell-1}} = E[z_{h_{\ell-1}}] \\ W_{h_\ell,h_{\ell-1}} = E[W_{h_\ell,h_{\ell-1}}]}} \\ &\bullet \begin{bmatrix} z_{h_{\ell-1}} - E[z_{h_{\ell-1}}] \\ W_{h_\ell,h_{\ell-1}} - E[W_{h_\ell,h_{\ell-1}}] \end{bmatrix} \Bigg]\end{aligned} \tag{42}$$

Because $E[z_{h_{\ell-1}} - E[z_{h_{\ell-1}}]] = 0$, $E[W_{h_\ell,h_{\ell-1}} - E[W_{h_\ell,h_{\ell-1}}]] = 0$, and $Cov[z_{h_{\ell-1}}, W_{h_\ell,h_{\ell-1}}] = 0$. This simplifies to:

$$
\begin{aligned}
&E[g(z_{h_{\ell-1}}, W_{h_\ell,h_{\ell-1}})] \approx g(E[z_{h_{\ell-1}}], E[W_{h_\ell,h_{\ell-1}}]) \\
&+ \frac{1}{2} Var \begin{bmatrix} z_{h_{\ell-1}} \\ W_{h_\ell,h_{\ell-1}} \end{bmatrix}^T \begin{bmatrix} \frac{\partial^2 g(z_{h_{\ell-1}}, W_{h_\ell,h_{\ell-1}})}{\partial^2 z_{h_{\ell-1}}} \\ \frac{\partial^2 g(z_{h_{\ell-1}}, W_{h_\ell,h_{\ell-1}})}{\partial^2 W_{h_\ell,h_{\ell-1}}} \end{bmatrix} \Bigg|_{\begin{cases} z_{h_{\ell-1}} = E[z_{h_{\ell-1}}] \\ W_{h_\ell,h_{\ell-1}} = E[W_{h_\ell,h_{\ell-1}}] \end{cases}}
\end{aligned} \tag{43}
$$

Inserting the derivatives and computing the inner products yields:

$$
\begin{aligned}
&E[g(z_{h_{\ell-1}}, W_{h_\ell,h_{\ell-1}})] \approx (|E[z_{h_{\ell-1}}]| + \epsilon)^{E[W_{h_\ell,h_{\ell-1}}]} \\
&+ \frac{1}{2} Var[z_{h_{\ell-1}}](|E[z_{h_{\ell-1}}]| + \epsilon)^{E[W_{h_\ell,h_{\ell-1}}]-2} E[W_{h_\ell,h_{\ell-1}}](E[W_{h_\ell,h_{\ell-1}}] - 1) \\
&+ \frac{1}{2} Var[W_{h_\ell,h_{\ell-1}}](|E[z_{h_{\ell-1}}]| + \epsilon)^{E[W_{h_\ell,h_{\ell-1}}]} \log(|E[z_{h_{\ell-1}}]| + \epsilon)^2 \\
&= 1 + \frac{1}{2} Var[W_{h_\ell,h_{\ell-1}}] \log(|E[z_{h_{\ell-1}}]| + \epsilon)^2
\end{aligned} \tag{44}
$$

This gives the final expectation:

$$
\begin{aligned}
E[z_{h_\ell}] &= E\left[g(z_{h_{\ell-1}}, W_{h_\ell,h_{\ell-1}})\right]^{H_{\ell-1}} \\
&\approx \left(1 + \frac{1}{2} Var[W_{h_\ell,h_{\ell-1}}] \log(|E[z_{h_{\ell-1}}]| + \epsilon)^2\right)^{H_{\ell-1}}
\end{aligned} \tag{45}
$$

As this expectation is of particular interest, we evaluate the error of the approximation, where $W_{h_\ell,h_{\ell-1}} \sim U[-r_w, r_w]$ and $z_{h_{\ell-1}} \sim U[0, r_z]$. These distributions are what is used in the arithmetic dataset. The error is plotted in figure 5.

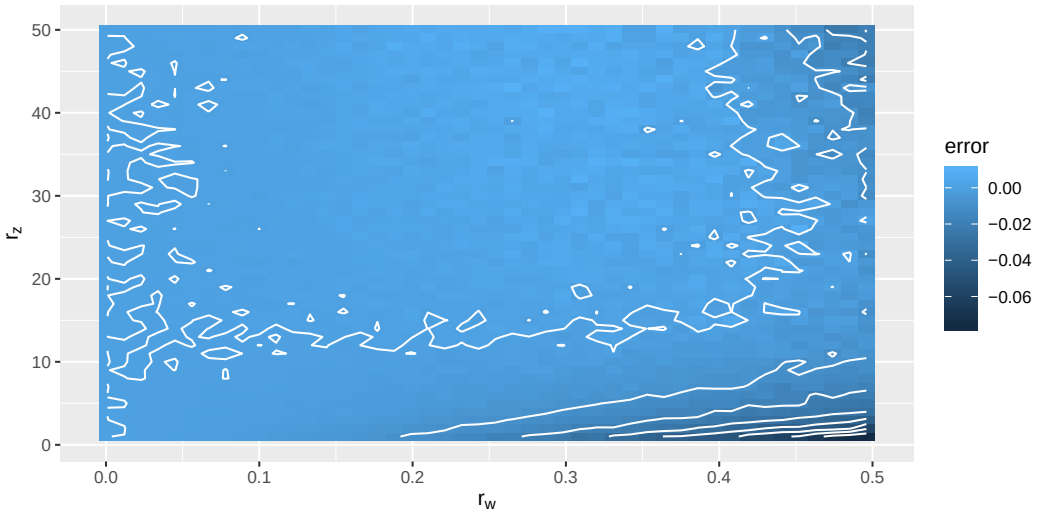


Figure 5: Error between theoretical approximation and the numerical approximation estimated by random sampling of 100000 observations at each combination of r_z and r_w.

Variance The variance can be derived using the same assumptions for expectation, that all $z_{h_{\ell-1}}$ are uncorrelated.

$$\begin{aligned} Var[z_{h_\ell}] &= E[z_{h_\ell}^2] - E[z_{h_\ell}]^2 \\ &= E\left[\prod_{h_{\ell-1}=1}^{H_{\ell-1}} (|z_{h_{\ell-1}}| + \epsilon)^{2 \cdot W_{h_\ell, h_{\ell-1}}}\right] - E\left[\prod_{h_{\ell-1}=1}^{H_{\ell-1}} (|z_{h_{\ell-1}}| + \epsilon)^{W_{h_\ell, h_{\ell-1}}}\right]^2 \\ &= E\left[g(z_{h_{\ell-1}}, 2 \cdot W_{h_\ell, h_{\ell-1}})\right]^{H_{\ell-1}} - E\left[g(z_{h_{\ell-1}}, W_{h_\ell, h_{\ell-1}})\right]^{2 \cdot H_{\ell-1}} \end{aligned} \tag{46}$$

We already have from the expectation result that:

$$E\left[g(z_{h_{\ell-1}}, W_{h_\ell, h_{\ell-1}})\right] \approx 1 + \frac{1}{2} Var[W_{h_\ell, h_{\ell-1}}] \log(|E[z_{h_{\ell-1}}]| + \epsilon)^2 \tag{47}$$

By substitution of variable we have that:

$$\begin{aligned} E\left[g(z_{h_{\ell-1}}, 2 \cdot W_{h_\ell, h_{\ell-1}})\right] &\approx 1 + \frac{1}{2} Var[2 \cdot W_{h_\ell, h_{\ell-1}}] \log(|E[z_{h_{\ell-1}}]| + \epsilon)^2 \\ &=\approx 1 + 2 \cdot Var[W_{h_\ell, h_{\ell-1}}] \log(|E[z_{h_{\ell-1}}]| + \epsilon)^2 \end{aligned} \tag{48}$$

This gives the variance:

$$\begin{aligned} Var[z_{h_\ell}] &= E\left[g(z_{h_{\ell-1}}, 2 \cdot W_{h_\ell, h_{\ell-1}})\right]^{H_{\ell-1}} - E\left[g(z_{h_{\ell-1}}, W_{h_\ell, h_{\ell-1}})\right]^{2 \cdot H_{\ell-1}} \\ &\approx \left(1 + 2 \cdot Var[W_{h_\ell, h_{\ell-1}}] \log(|E[z_{h_{\ell-1}}]| + \epsilon)^2\right)^{H_{\ell-1}} \\ &- \left(1 + \frac{1}{2} \cdot Var[W_{h_\ell, h_{\ell-1}}] \log(|E[z_{h_{\ell-1}}]| + \epsilon)^2\right)^{2 \cdot H_{\ell-1}} \end{aligned} \tag{49}$$

B.3.2 BACKWARD PASS

Expectation The expectation of the back-propagation term assuming that $\delta_{h_{\ell+1}}$ and $\frac{\partial z_{h_{\ell+1}}}{\partial z_{h_\ell}}$ are mutually uncorrelated:

$$E[\delta_{h_\ell}] = E\left[\sum_{h_{\ell+1}=1}^{H_{\ell+1}} \delta_{h_{\ell+1}} \frac{\partial z_{h_{\ell+1}}}{\partial z_{h_\ell}}\right] \approx H_{\ell+1} E[\delta_{h_{\ell+1}}] E\left[\frac{\partial z_{h_{\ell+1}}}{\partial z_{h_\ell}}\right] \tag{50}$$

Assuming that $z_{h_{\ell+1}}$, $W_{h_{\ell+1}, h_\ell}$, and z_{h_ℓ} are uncorrelated:

$$E\left[\frac{\partial z_{h_{\ell+1}}}{\partial z_{h_\ell}}\right] \approx E[z_{h_{\ell+1}}] E[W_{h_{\ell+1}, h_\ell}] E\left[\frac{\text{abs}'(z_{h_\ell})}{|z_{h_\ell}| + \epsilon}\right] = E[z_{h_{\ell+1}}] \cdot 0 \cdot E\left[\frac{\text{abs}'(z_{h_\ell})}{|z_{h_\ell}| + \epsilon}\right] = 0 \tag{51}$$

Variance Deriving the variance is more complicated as:

$$Var\left[\frac{\partial z_{h_{\ell+1}}}{\partial z_{h_\ell}}\right] = Var\left[z_{h_{\ell+1}} W_{h_{\ell+1}, h_\ell} \frac{\text{abs}'(z_{h_\ell})}{|z_{h_\ell}| + \epsilon}\right] \tag{52}$$

Assuming again that $z_{h_{\ell+1}}$, $W_{h_{\ell+1}, h_\ell}$, and z_{h_ℓ} are uncorrelated, and likewise for their second moment:

$$\begin{aligned} Var\left[\frac{\partial z_{h_{\ell+1}}}{\partial z_{h_\ell}}\right] &\approx E[z_{h_{\ell+1}}^2] E[W_{h_{\ell+1}, h_\ell}^2] E\left[\left(\frac{\text{abs}'(z_{h_\ell})}{|z_{h_\ell}| + \epsilon}\right)^2\right] \\ &- E[z_{h_{\ell+1}}]^2 E[W_{h_{\ell+1}, h_\ell}]^2 E\left[\frac{\text{abs}'(z_{h_\ell})}{|z_{h_\ell}| + \epsilon}\right]^2 \\ &= E[z_{h_{\ell+1}}^2] Var[W_{h_{\ell+1}, h_\ell}] E\left[\left(\frac{\text{abs}'(z_{h_\ell})}{|z_{h_\ell}| + \epsilon}\right)^2\right] \\ &- E[z_{h_{\ell+1}}]^2 \cdot 0 \cdot E\left[\frac{\text{abs}'(z_{h_\ell})}{|z_{h_\ell}| + \epsilon}\right]^2 \\ &= E[z_{h_{\ell+1}}^2] Var[W_{h_{\ell+1}, h_\ell}] E\left[\left(\frac{\text{abs}'(z_{h_\ell})}{|z_{h_\ell}| + \epsilon}\right)^2\right] \end{aligned} \tag{53}$$

Using Taylor approximation around $E[z_{h_\ell}]$ we have:

$$\begin{aligned} E\left[\left(\frac{\text{abs}'(z_{h_\ell})}{|z|+\epsilon}\right)^2\right] &\approx \frac{1}{\left(|E[z_{h_\ell}]|+\epsilon\right)^2} + \frac{1}{2}\frac{6}{\left(|E[z_{h_\ell}]|+\epsilon\right)^4} Var[z_{h_\ell}] \\ &= \frac{1}{\left(|E[z_{h_\ell}]|+\epsilon\right)^2} + \frac{3}{\left(|E[z_{h_\ell}]|+\epsilon\right)^4} Var[z_{h_\ell}] \end{aligned} \tag{54}$$

Finally, by reusing the result for $E[z_{h_\ell}^2]$ from earlier the variance can be expressed as:

$$\begin{aligned} Var\left[\frac{\partial \mathcal{L}}{\partial z_{h_{\ell-1}}}\right] \approx Var\left[\frac{\partial \mathcal{L}}{\partial z_{h_\ell}}\right] H_\ell \ \left(1 + 2 \cdot Var[W_{h_\ell,h_{\ell-1}}] \log(|E[z_{h_{\ell-1}}]|+\epsilon)^2\right)^{H_{\ell-1}} \\ \cdot Var[W_{h_\ell,h_{\ell-1}}] \left(\frac{1}{\left(|E[z_{h_{\ell-1}}]|+\epsilon\right)^2} + \frac{3}{\left(|E[z_{h_{\ell-1}}]|+\epsilon\right)^4} Var[z_{h_{\ell-1}}]\right) \end{aligned} \tag{55}$$

B.4 Expectation and variance of NMU

B.4.1 Forward pass

Expectation Assuming that all $z_{h_{\ell-1}}$ are independent:

$$\begin{aligned} E[z_{h_\ell}] &= E\left[\prod_{h_{\ell-1}=1}^{H_{\ell-1}} \left(W_{h_{\ell-1},h_\ell} z_{h_{\ell-1}} + 1 - W_{h_{\ell-1},h_\ell}\right)\right] \\ &\approx E\left[W_{h_{\ell-1},h_\ell} z_{h_{\ell-1}} + 1 - W_{h_{\ell-1},h_\ell}\right]^{H_{\ell-1}} \\ &\approx \left(E[W_{h_{\ell-1},h_\ell}] E[z_{h_{\ell-1}}] + 1 - E[W_{h_{\ell-1},h_\ell}]\right)^{H_{\ell-1}} \end{aligned} \tag{56}$$

Assuming that $E[z_{h_{\ell-1}}] = 0$ which is a desired property and initializing $E[W_{h_{\ell-1},h_\ell}] = 1/2$, the expectation is:

$$\begin{aligned} E[z_{h_\ell}] &\approx \left(E[W_{h_{\ell-1},h_\ell}] E[z_{h_{\ell-1}}] + 1 - E[W_{h_{\ell-1},h_\ell}]\right)^{H_{\ell-1}} \\ &\approx \left(\frac{1}{2} \cdot 0 + 1 - \frac{1}{2}\right)^{H_{\ell-1}} \\ &= \left(\frac{1}{2}\right)^{H_{\ell-1}} \end{aligned} \tag{57}$$

Variance Reusing the result for the expectation, assuming again that all $z_{h_{\ell-1}}$ are uncorrelated, and using the fact that $W_{h_{\ell-1},h_\ell}$ is initially independent from $z_{h_{\ell-1}}$:

$$\begin{aligned} Var[z_{h_\ell}] &= E[z_{h_\ell}^2] - E[z_{h_\ell}]^2 \\ &\approx E[z_{h_\ell}^2] - \left(\frac{1}{2}\right)^{2 \cdot H_{\ell-1}} \\ &= E\left[\prod_{h_{\ell-1}=1}^{H_{\ell-1}} \left(W_{h_{\ell-1},h_\ell} z_{h_{\ell-1}} + 1 - W_{h_{\ell-1},h_\ell}\right)^2\right] - \left(\frac{1}{2}\right)^{2 \cdot H_{\ell-1}} \\ &\approx E[\left(W_{h_{\ell-1},h_\ell} z_{h_{\ell-1}} + 1 - W_{h_{\ell-1},h_\ell}\right)^2]^{H_{\ell-1}} - \left(\frac{1}{2}\right)^{2 \cdot H_{\ell-1}} \\ &= \Big(E[W_{h_{\ell-1},h_\ell}^2] E[z_{h_{\ell-1}}^2] - 2E[W_{h_{\ell-1},h_\ell}^2] E[z_{h_{\ell-1}}] + E[W_{h_{\ell-1},h_\ell}^2] \\ &\quad + 2E[W_{h_{\ell-1},h_\ell}] E[z_{h_{\ell-1}}] - 2E[W_{h_{\ell-1},h_\ell}] + 1\Big)^{H_{\ell-1}} - \left(\frac{1}{2}\right)^{2 \cdot H_{\ell-1}} \end{aligned} \tag{58}$$

Assuming again that $E[z_{h_{\ell-1}}] = 0$, which is a desired property and initializing $E[W_{h_{\ell-1},h_\ell}] = 1/2$, the variance becomes:

$$\begin{aligned} Var[z_{h_\ell}] &\approx \left(E[W_{h_{\ell-1},h_\ell}^2]\left(E[z_{h_{\ell-1}}^2]+1\right)\right)^{H_{\ell-1}} - \left(\frac{1}{2}\right)^{2\cdot H_{\ell-1}} \\ &\approx \left(\left(Var[W_{h_{\ell-1},h_\ell}] + E[W_{h_{\ell-1},h_\ell}]^2\right)\left(Var[z_{h_{\ell-1}}]+1\right)\right)^{H_{\ell-1}} - \left(\frac{1}{2}\right)^{2\cdot H_{\ell-1}} \\ &= \left(Var[W_{h_{\ell-1},h_\ell}] + \frac{1}{4}\right)^{H_{\ell-1}}\left(Var[z_{h_{\ell-1}}]+1\right)^{H_{\ell-1}} - \left(\frac{1}{2}\right)^{2\cdot H_{\ell-1}} \end{aligned} \tag{59}$$

B.4.2 BACKWARD PASS

Expectation For the backward pass the expectation can, assuming that $\frac{\partial \mathcal{L}}{\partial z_{h_\ell}}$ and $\frac{\partial z_{h_\ell}}{\partial z_{h_{\ell-1}}}$ are uncorrelated, be derived to:

$$\begin{aligned} E\left[\frac{\partial \mathcal{L}}{\partial z_{h_{\ell-1}}}\right] &= H_\ell E\left[\frac{\partial \mathcal{L}}{\partial z_{h_\ell}}\frac{\partial z_{h_\ell}}{\partial z_{h_{\ell-1}}}\right] \\ &\approx H_\ell E\left[\frac{\partial \mathcal{L}}{\partial z_{h_\ell}}\right] E\left[\frac{\partial z_{h_\ell}}{\partial z_{h_{\ell-1}}}\right] \\ &= H_\ell E\left[\frac{\partial \mathcal{L}}{\partial z_{h_\ell}}\right] E\left[\frac{z_{h_\ell}}{W_{h_{\ell-1},h_\ell} z_{h_{\ell-1}} + 1 - W_{h_{\ell-1},h_\ell}} W_{h_{\ell-1},h_\ell}\right] \\ &= H_\ell E\left[\frac{\partial \mathcal{L}}{\partial z_{h_\ell}}\right] E\left[\frac{z_{h_\ell}}{W_{h_{\ell-1},h_\ell} z_{h_{\ell-1}} + 1 - W_{h_{\ell-1},h_\ell}}\right] E\left[W_{h_{\ell-1},h_\ell}\right] \end{aligned} \tag{60}$$

Initializing again $E[W_{h_{\ell-1},h_\ell}] = 1/2$, and inserting the result for the expectation $E\left[\frac{z_{h_\ell}}{W_{h_{\ell-1},h_\ell} z_{h_{\ell-1}} + 1 - W_{h_{\ell-1},h_\ell}}\right]$.

$$\begin{aligned} E\left[\frac{\partial \mathcal{L}}{\partial z_{h_{\ell-1}}}\right] &\approx H_\ell E\left[\frac{\partial \mathcal{L}}{\partial z_{h_\ell}}\right]\left(\frac{1}{2}\right)^{H_{\ell-1}-1}\frac{1}{2} \\ &= E\left[\frac{\partial \mathcal{L}}{\partial z_{h_\ell}}\right] H_\ell \left(\frac{1}{2}\right)^{H_{\ell-1}} \end{aligned} \tag{61}$$

Assuming that $E\left[\frac{\partial \mathcal{L}}{\partial z_{h_\ell}}\right] = 0$, which is a desired property (**?**).

$$\begin{aligned} E\left[\frac{\partial \mathcal{L}}{\partial z_{h_{\ell-1}}}\right] &\approx 0 \cdot H_\ell \cdot \left(\frac{1}{2}\right)^{H_{\ell-1}} \\ &= 0 \end{aligned} \tag{62}$$

Variance For the variance of the backpropagation term, we assume that $\frac{\partial \mathcal{L}}{\partial z_{h_\ell}}$ is uncorrelated with $\frac{\partial z_{h_\ell}}{\partial z_{h_{\ell-1}}}$.

$$\begin{aligned} Var\left[\frac{\partial \mathcal{L}}{\partial z_{h_{\ell-1}}}\right] &= H_\ell Var\left[\frac{\partial \mathcal{L}}{\partial z_{h_\ell}}\frac{\partial z_{h_\ell}}{\partial z_{h_{\ell-1}}}\right] \\ &\approx H_\ell \Bigg(Var\left[\frac{\partial \mathcal{L}}{\partial z_{h_\ell}}\right] E\left[\frac{\partial z_{h_\ell}}{\partial z_{h_{\ell-1}}}\right]^2 + E\left[\frac{\partial \mathcal{L}}{\partial z_{h_\ell}}\right]^2 Var\left[\frac{\partial z_{h_\ell}}{\partial z_{h_{\ell-1}}}\right] \\ &\quad + Var\left[\frac{\partial \mathcal{L}}{\partial z_{h_\ell}}\right] Var\left[\frac{\partial z_{h_\ell}}{\partial z_{h_{\ell-1}}}\right] \Bigg) \end{aligned} \tag{63}$$

Assuming again that $E\left[\frac{\partial \mathcal{L}}{\partial z_{h_\ell}}\right] = 0$, and reusing the result $E\left[\frac{\partial z_{h_\ell}}{\partial z_{h_{\ell-1}}}\right] = \left(\frac{1}{2}\right)^{H_{\ell-1}}$.

$$Var\left[\frac{\partial \mathcal{L}}{\partial z_{h_{\ell-1}}}\right] \approx Var\left[\frac{\partial \mathcal{L}}{\partial z_{h_\ell}}\right] H_\ell \left(\left(\frac{1}{2}\right)^{2 \cdot H_{\ell-1}} + Var\left[\frac{\partial z_{h_\ell}}{\partial z_{h_{\ell-1}}}\right]\right) \tag{64}$$

Focusing now on $Var\left[\frac{\partial z_{h_\ell}}{\partial z_{h_{\ell-1}}}\right]$, we have:

$$\begin{aligned} Var\left[\frac{\partial z_{h_\ell}}{\partial z_{h_{\ell-1}}}\right] &= E\left[\left(\frac{z_{h_\ell}}{W_{h_{\ell-1},h_\ell} z_{h_{\ell-1}} + 1 - W_{h_{\ell-1},h_\ell}}\right)^2\right] E[W^2_{h_{\ell-1},h_\ell}] \\ &\quad - E\left[\frac{z_{h_\ell}}{W_{h_{\ell-1},h_\ell} z_{h_{\ell-1}} + 1 - W_{h_{\ell-1},h_\ell}}\right]^2 E[W_{h_{\ell-1},h_\ell}]^2 \end{aligned} \tag{65}$$

Inserting the result for the expectation $E\left[\frac{z_{h_\ell}}{W_{h_{\ell-1},h_\ell} z_{h_{\ell-1}} + 1 - W_{h_{\ell-1},h_\ell}}\right]$ and Initializing again $E[W_{h_{\ell-1},h_\ell}] = 1/2$.

$$\begin{aligned} Var\left[\frac{\partial z_{h_\ell}}{\partial z_{h_{\ell-1}}}\right] &\approx E\left[\left(\frac{z_{h_\ell}}{W_{h_{\ell-1},h_\ell} z_{h_{\ell-1}} + 1 - W_{h_{\ell-1},h_\ell}}\right)^2\right] E[W^2_{h_{\ell-1},h_\ell}] \\ &\quad - \left(\frac{1}{2}\right)^{2 \cdot (H_{\ell-1}-1)} \left(\frac{1}{2}\right)^2 \\ &= E\left[\left(\frac{z_{h_\ell}}{W_{h_{\ell-1},h_\ell} z_{h_{\ell-1}} + 1 - W_{h_{\ell-1},h_\ell}}\right)^2\right] E[W^2_{h_{\ell-1},h_\ell}] \\ &\quad - \left(\frac{1}{2}\right)^{2 \cdot H_{\ell-1}} \end{aligned} \tag{66}$$

Using the identity that $E[W^2_{h_{\ell-1},h_\ell}] = Var[W_{h_{\ell-1},h_\ell}] + E[W_{h_{\ell-1},h_\ell}]^2$, and again using $E[W_{h_{\ell-1},h_\ell}] = 1/2$.

$$\begin{aligned} Var\left[\frac{\partial z_{h_\ell}}{\partial z_{h_{\ell-1}}}\right] &\approx E\left[\left(\frac{z_{h_\ell}}{W_{h_{\ell-1},h_\ell} z_{h_{\ell-1}} + 1 - W_{h_{\ell-1},h_\ell}}\right)^2\right] \left(Var[W_{h_{\ell-1},h_\ell}] + \frac{1}{4}\right) \\ &\quad - \left(\frac{1}{2}\right)^{2 \cdot H_{\ell-1}} \end{aligned} \tag{67}$$

To derive $E\left[\left(\frac{z_{h_\ell}}{W_{h_{\ell-1},h_\ell} z_{h_{\ell-1}} + 1 - W_{h_{\ell-1},h_\ell}}\right)^2\right]$ the result for $Var[z_{h_\ell}]$ can be used, but for $\hat{H}_{\ell-1} = H_{\ell-1} - 1$, because there is one less term. Inserting $E\left[\left(\frac{z_{h_\ell}}{W_{h_{\ell-1},h_\ell} z_{h_{\ell-1}} + 1 - W_{h_{\ell-1},h_\ell}}\right)^2\right] = \left(Var[W_{h_{\ell-1},h_\ell}] + \frac{1}{4}\right)^{H_{\ell-1}-1} \left(Var[z_{h_{\ell-1}}] + 1\right)^{H_{\ell-1}-1}$, we have:

$$\begin{aligned} Var\left[\frac{\partial z_{h_\ell}}{\partial z_{h_{\ell-1}}}\right] &\approx \left(Var[W_{h_{\ell-1},h_\ell}] + \frac{1}{4}\right)^{H_{\ell-1}-1} \left(Var[z_{h_{\ell-1}}] + 1\right)^{H_{\ell-1}-1} \\ &\quad \cdot \left(Var[W_{h_{\ell-1},h_\ell}] + \frac{1}{4}\right) - \left(\frac{1}{2}\right)^{2 \cdot H_{\ell-1}} \\ &= \left(Var[W_{h_{\ell-1},h_\ell}] + \frac{1}{4}\right)^{H_{\ell-1}} \left(Var[z_{h_{\ell-1}}] + 1\right)^{H_{\ell-1}-1} - \left(\frac{1}{2}\right)^{2 \cdot H_{\ell-1}} \end{aligned} \tag{68}$$

Inserting the result for $Var\left[\frac{\partial z_{h_\ell}}{\partial z_{h_{\ell-1}}}\right]$ into the result for $Var\left[\frac{\partial \mathcal{L}}{\partial z_{h_{\ell-1}}}\right]$:

$$\begin{aligned} Var\left[\frac{\partial \mathcal{L}}{\partial z_{h_{\ell-1}}}\right] \approx Var\left[\frac{\partial \mathcal{L}}{\partial z_{h_\ell}}\right] H_\ell \Bigg(& \left(\frac{1}{2}\right)^{2 \cdot H_{\ell-1}} \\ & + \left(Var[W_{h_{\ell-1},h_\ell}] + \frac{1}{4}\right)^{H_{\ell-1}} \left(Var[z_{h_{\ell-1}}] + 1\right)^{H_{\ell-1}-1} - \left(\frac{1}{2}\right)^{2 \cdot H_{\ell-1}} \Bigg) \\ & = Var\left[\frac{\partial \mathcal{L}}{\partial z_{h_\ell}}\right] H_\ell \\ & \cdot \left(\left(Var[W_{h_{\ell-1},h_\ell}] + \frac{1}{4}\right)^{H_{\ell-1}} \left(Var[z_{h_{\ell-1}}] + 1\right)^{H_{\ell-1}-1}\right) \end{aligned} \tag{69}$$

B.4.3 INITIALIZATION

The $W_{h_{\ell-1},h_\ell}$ should be initialized with $E[W_{h_{\ell-1},h_\ell}] = \frac{1}{2}$, in order to not bias towards inclusion or exclusion of $z_{h_{\ell-1}}$. Using the derived variance approximations, the variance should be according to the forward pass:

$$Var[W_{h_{\ell-1},h_\ell}] = \left((1 + Var[z_{h_\ell}])^{-H_{\ell-1}} Var[z_{h_\ell}] + (4 + 4Var[z_{h_\ell}])^{-H_{\ell-1}}\right)^{\frac{1}{H_{\ell-1}}} - \frac{1}{4} \tag{70}$$

And according to the backward pass it should be:

$$Var[W_{h_{\ell-1},h_\ell}] = \left(\frac{(Var[z_{h_\ell}] + 1)^{1-H_{\ell-1}}}{H_\ell}\right)^{\frac{1}{H_{\ell-1}}} - \frac{1}{4} \tag{71}$$

Both criteria are dependent on the input variance. If the input variance is know then optimal initialization is possible. However, as this is often not the case one can perhaps assume that $Var[z_{h_{\ell-1}}] = 1$. This is not an unreasonable assumption in many cases, as there may either be a normalization layer somewhere or the input is normalized. If unit variance is assumed, one gets from the forward pass:

$$Var[W_{h_{\ell-1},h_\ell}] = \left(2^{-H_{\ell-1}} + 8^{-H_{\ell-1}}\right)^{\frac{1}{H_{\ell-1}}} - \frac{1}{4} = \frac{1}{8}\left(\left(4^{H_{\ell-1}} + 1\right)^{H_{\ell-1}} - 2\right) \tag{72}$$

And from the backward pass:

$$Var[W_{h_{\ell-1},h_\ell}] = \left(\frac{2^{1-H_{\ell-1}}}{H_\ell}\right)^{\frac{1}{H_{\ell-1}}} - \frac{1}{4} \tag{73}$$

The variance requirement for both the forward and backward pass can be satisfied with $Var[W_{h_{\ell-1},h_\ell}] = \frac{1}{4}$ for a large $H_{\ell-1}$.

C SIMPLE FUNCTION TASK

C.1 DATASET GENERATION

We define the values a and b, as the sub of subsets of $\mathbf{x}$ (see figure 6). The exact dataset generation algorithm is defined in algorithm 1.

$$a = \sum_{i=s_{1,\text{start}}}^{s_{1,\text{end}}} x_i, \quad b = \sum_{i=s_{2,\text{start}}}^{s_{2,\text{end}}} x_i \tag{74}$$

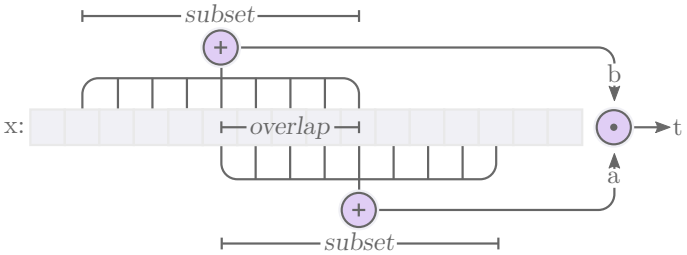


Figure 6: Dataset is parameterized into "Input Size", "Subset Ratio", "Overlap Ratio", an Operation (here showing multiplication), "Interpolation Range" and "Extrapolation Range" from which the data set sampled.

Algorithm 1 Dataset sampling algorithm

function DATASET(OP$(\cdot,\cdot)$: Operation, i : InputSize, s : SubsetRatio, o : OverlapRatio, R : Range)
  $\mathbf{x} \leftarrow$ UNIFORM$(R_{lower}, R_{upper}, i)$ ▷ Sample i elements uniformly
  $k \leftarrow$ UNIFORM$(0, 1 - 2s - o)$ ▷ Sample offset
  $a \leftarrow$ SUM$(\mathbf{x}[ik : i(k+s)])$ ▷ Create sum a from subset
  $b \leftarrow$ SUM$(\mathbf{x}[i(k+s-o) : i(k+2s-0)])$ ▷ Create sum b from subset
  $t \leftarrow$ OP(a, b) ▷ Perform operation on a and b
  return x, t

C.2 ARITHMETIC OPERATIONS COMPARISON - ALL MODELS

Models are defined in table 3 and are all optimized with Adam optimization (**?**) using default parameters and trained for about four hours on an HPC cluster running 8-Core Intel Xeon E5-2665 2.4GHz CPUs.

The training dataset is continuously sampled from the interpolation range where a different seed is used for each experiment, all experiments has a mini-batch size of 128 observations, a fixed validationset with $1 \cdot 10^4$ observations sampled from the interpolation range, and a fixed testset with $1 \cdot 10^4$ observations sampled from the extrapolation range.

Results for all models on addition, subtraction, and multiplication can be found in table 4.

C.3 EFFECT OF DATASET PARAMETER

To stress test the models on the multiplication task, we vary the dataset parameters one at a time (table 5), and run 25 experiments with different seeds. The results, are visualized in figure 7.

C.4 ABLATION STUDY

To validate our model, we perform an ablation on the multiplication problem.

Table 3: Model definitions

Model	Layer 1	Layer 2	$\hat{\lambda}_{\text{sparse}}$	λ_{start}	λ_{end}
NMU	NAU	NMU	10	10^6	$2 \cdot 10^6$
NAU	NAU	NAU	0.01	$5 \cdot 10^3$	$5 \cdot 10^4$
$\text{NAC}_\bullet$	NAC_+	$\text{NAC}_\bullet$	–	–	–
$\text{NAC}_{\bullet,\sigma}$	NAC_+	$\text{NAC}_{\bullet,\sigma}$	–	–	–
$\text{NAC}_{\bullet,\text{NMU}}$	NAC_+	$\text{NAC}_{\bullet,\text{NMU}}$	10	10^6	$2 \cdot 10^6$
NAC_+	NAC_+	NAC_+	–	–	–
NALU	NALU	NALU	–	–	–
Linear	Linear	Linear	–	–	–
ReLU	ReLU	ReLU	–	–	–
ReLU6	ReLU6	ReLU6	–	–	–

Table 4: Shows the success-rate for $\mathcal{L}_{\mathbf{W}_1,\mathbf{W}_2} < \mathcal{L}_{\mathbf{W}_1^\epsilon,\mathbf{W}_2^*}$, at what global step the model converged at and the sparsity error for all weight matrices, with 95% confidence interval. Best result is highlighted without considering significance.

Op	Model	Success	Solved at		Sparsity error
		Rate	Median	Mean	Mean
$\times$	$\text{NAC}_{\bullet,\text{NMU}}$	96%	$2.0 \cdot 10^6$	$2.2 \cdot 10^6 \pm 3.6 \cdot 10^5$	$5.1 \cdot 10^{-7} \pm 8.2 \cdot 10^{-8}$
	$\text{NAC}_{\bullet,\sigma}$	**100%**	$2.5 \cdot 10^6$	$2.7 \cdot 10^6 \pm 2.1 \cdot 10^5$	$1.0 \cdot 10^{-4} \pm 4.9 \cdot 10^{-5}$
	$\text{NAC}_\bullet$	24%	$2.9 \cdot 10^6$	$3.0 \cdot 10^6 \pm 6.6 \cdot 10^5$	$4.0 \cdot 10^{-4} \pm 4.1 \cdot 10^{-4}$
	NAC_+	0%	—	—	—
	Linear	0%	—	—	—
	NALU	0%	—	—	—
	NAU	0%	—	—	—
	NMU	**100%**	$\mathbf{1.5 \cdot 10^6}$	$\mathbf{1.7 \cdot 10^6 \pm 3.1 \cdot 10^5}$	$\mathbf{4.4 \cdot 10^{-7} \pm 6.9 \cdot 10^{-8}}$
	ReLU	0%	—	—	—
	ReLU6	0%	—	—	—
$+$	$\text{NAC}_{\bullet,\text{NMU}}$	0%	—	—	—
	$\text{NAC}_{\bullet,\sigma}$	0%	—	—	—
	$\text{NAC}_\bullet$	0%	—	—	—
	NAC_+	**100%**	$2.4 \cdot 10^5$	$2.6 \cdot 10^5 \pm 4.9 \cdot 10^4$	$4.7 \cdot 10^{-1} \pm 2.4 \cdot 10^{-2}$
	Linear	**100%**	$5.8 \cdot 10^4$	$\mathbf{6.0 \cdot 10^4 \pm 4.2 \cdot 10^3}$	$5.5 \cdot 10^{-1} \pm 4.1 \cdot 10^{-2}$
	NALU	4%	$1.8 \cdot 10^6$	$1.8 \cdot 10^6$	$4.9 \cdot 10^{-1}$
	NAU	**100%**	$\mathbf{1.8 \cdot 10^4}$	$1.8 \cdot 10^5 \pm 2.4 \cdot 10^5$	$\mathbf{3.2 \cdot 10^{-5} \pm 2.8 \cdot 10^{-5}}$
	NMU	0%	—	—	—
	ReLU	72%	$6.7 \cdot 10^4$	$2.6 \cdot 10^5 \pm 2.9 \cdot 10^5$	$6.8 \cdot 10^{-1} \pm 1.1 \cdot 10^{-1}$
	ReLU6	0%	—	—	—
$-$	$\text{NAC}_{\bullet,\text{NMU}}$	0%	—	—	—
	$\text{NAC}_{\bullet,\sigma}$	0%	—	—	—
	$\text{NAC}_\bullet$	0%	—	—	—
	NAC_+	**100%**	$9.0 \cdot 10^3$	$6.9 \cdot 10^5 \pm 5.2 \cdot 10^5$	$4.5 \cdot 10^{-1} \pm 2.5 \cdot 10^{-2}$
	Linear	4%	$2.4 \cdot 10^6$	$2.4 \cdot 10^6$	$4.8 \cdot 10^{-1}$
	NALU	8%	$2.3 \cdot 10^6$	$2.3 \cdot 10^6 \pm 1.1 \cdot 10^7$	$4.7 \cdot 10^{-1} \pm 3.2 \cdot 10^{-1}$
	NAU	**100%**	$\mathbf{5.0 \cdot 10^3}$	$\mathbf{1.8 \cdot 10^5 \pm 3.6 \cdot 10^5}$	$1.8 \cdot 10^{-1} \pm 1.0 \cdot 10^{-1}$
	NMU	56%	$1.0 \cdot 10^6$	$1.0 \cdot 10^6 \pm 6.1 \cdot 10^2$	$\mathbf{6.4 \cdot 10^{-4} \pm 1.0 \cdot 10^{-4}}$
	ReLU	0%	—	—	—
	ReLU6	0%	—	—	—

Our ablation study (table 6) show that regularization have little effect in terms of success rate. As it is analytically known that there is no gradient outside of $w \in [0, 1]$ for the NMU, the conclusion must be that the optimal weight initialization for the default dataset parameters and tested seeds, does not cause any weights to accidentally break out of $w \in [0, 1]$. The sparse regularizer for multiplication

Table 5: Default dataset parameters

Parameter name	Default value
Input size	100
Subset ratio	0.25
Overlap ratio	0.5

Parameter name	Default value
Interpolation range	$U[1, 2]$
Extrapolation range	$U[2, 6]$

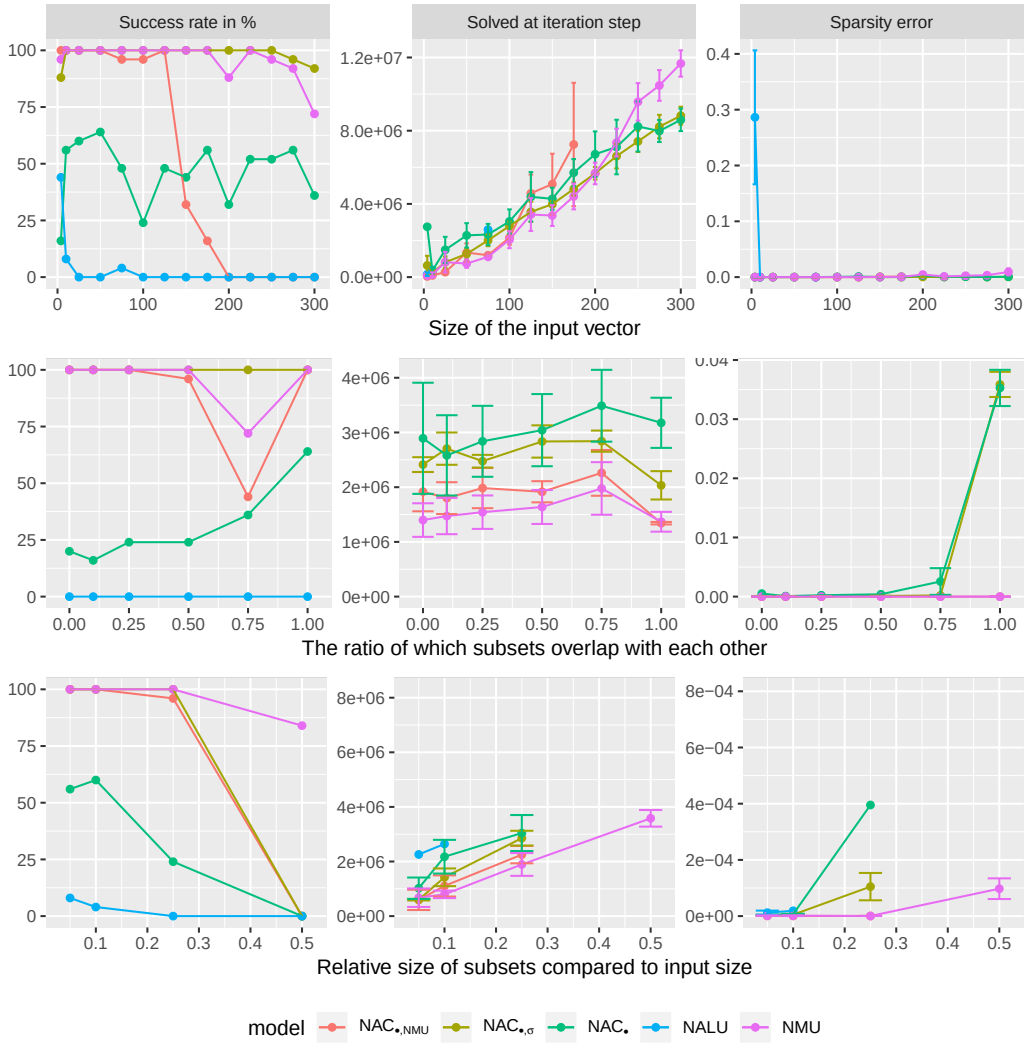


Figure 7: Shows the effect of the dataset parameters.

have no sparsity effect, as only a sparse solution is a valid solution for multiplication. Although as seen in appendix C.5, sparsity regularization can improve convergence.

Not allowing a multiplicative identity ($\mathbf{z} = \mathbf{W} \odot \mathbf{x}$, instead of $\mathbf{z} = \mathbf{W} \odot \mathbf{x} + 1 - \mathbf{W}$), works when there is only two hidden units in the multiplication layer, as no multiplicative identity is necessary. However, for a larger hidden size, as seen in figure 8, multiplicative identity becomes necessary.

missing figure

Figure 8: Compares NMU with NMU without identity, with different input size (hidden layer unit size) to the multiplication layer.

Table 6: Shows the success-rate for $\mathcal{L}_{\mathbf{W}_1,\mathbf{W}_2} < \mathcal{L}_{\mathbf{W}_1^\epsilon,\mathbf{W}_2^*}$, at what global step the model converged at and the sparsity error for all weight matrices, with 95% confidence interval. The dataset is the multiplication problem with default parameters.

Model	Success	Solved at		Sparsity error
	Rate	Median	Mean	Mean
$\text{NAC}_{\bullet,\sigma}$	100%	$2.5 \cdot 10^6$	$2.6 \cdot 10^6 \pm 1.6 \cdot 10^5$	$1.0 \cdot 10^{-4} \pm 4.9 \cdot 10^{-5}$
$\text{NAC}_{\bullet}$	24%	$2.9 \cdot 10^6$	$3.0 \cdot 10^6 \pm 6.6 \cdot 10^5$	$4.0 \cdot 10^{-4} \pm 4.1 \cdot 10^{-4}$
NMU	100%	$1.4 \cdot 10^6$	$1.4 \cdot 10^6 \pm 6.7 \cdot 10^4$	$4.4 \cdot 10^{-7} \pm 6.9 \cdot 10^{-8}$
NMU, no $\mathcal{R}_{bias}$	100%	$1.9 \cdot 10^6$	$1.9 \cdot 10^6 \pm 1.2 \cdot 10^5$	$9.4 \cdot 10^{-4} \pm 2.9 \cdot 10^{-4}$
NMU, no $\mathcal{R}_{bias}$, no W-clamp	100%	$1.5 \cdot 10^6$	$1.5 \cdot 10^6 \pm 1.0 \cdot 10^5$	$2.8 \cdot 10^{-4} \pm 5.3 \cdot 10^{-5}$
NMU, no W-clamp	100%	$1.3 \cdot 10^6$	$1.3 \cdot 10^6 \pm 6.6 \cdot 10^4$	$8.8 \cdot 10^{-5} \pm 6.2 \cdot 10^{-5}$

C.5 REGULARIZATION

A high sparsity regularization constant can help the model to converge faster. However, a regularization constant too high have have the inverse effect as well, or even make it impossible for the model to converge.

In these experiments, the constant c in equation 75 is varied. See results in figure 9, 10, and 11.

$$\lambda_{\text{bias}} = c \cdot (1 - \exp(-10^5 \cdot t)) \tag{75}$$

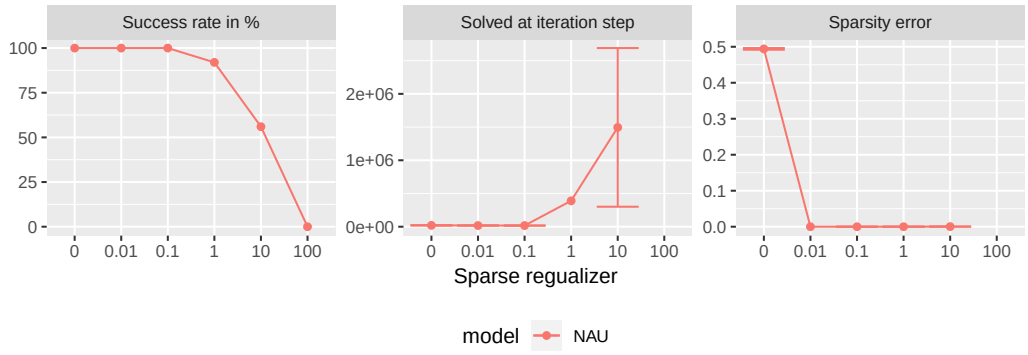


Figure 9: Shows the effect of the regularizer parameter c in λ_{bias}, on the arithmetic dataset for the $+$ operation.

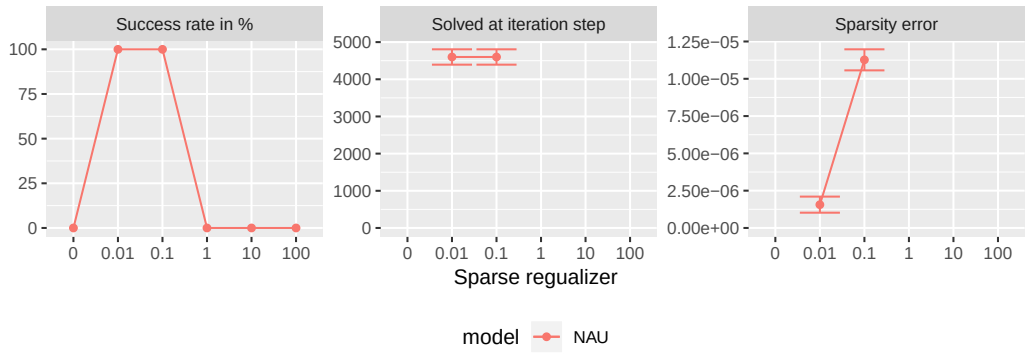


Figure 10: Shows the effect of the regularizer parameter c in λ_{bias}, on the arithmetic dataset for the $-$ operation.

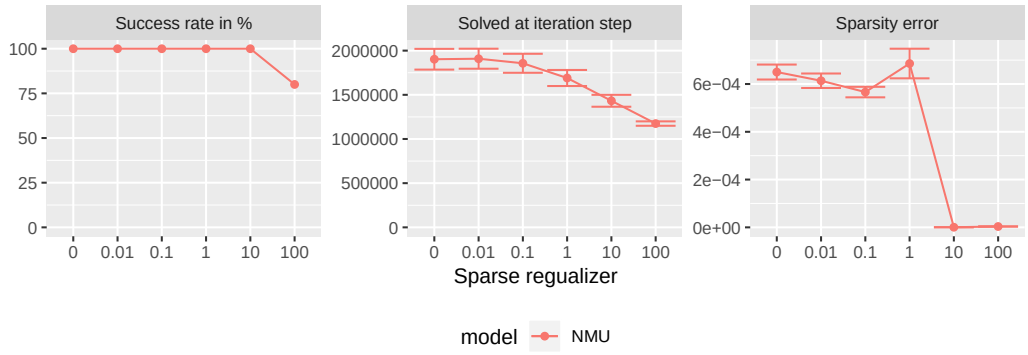


Figure 11: Shows the effect of the regularizer parameter c in λ_{bias}, on the arithmetic dataset for the $\times$ operation.

D SEQUENTIAL MNIST

D.1 WITHOUT THE R_z REGULARIZER

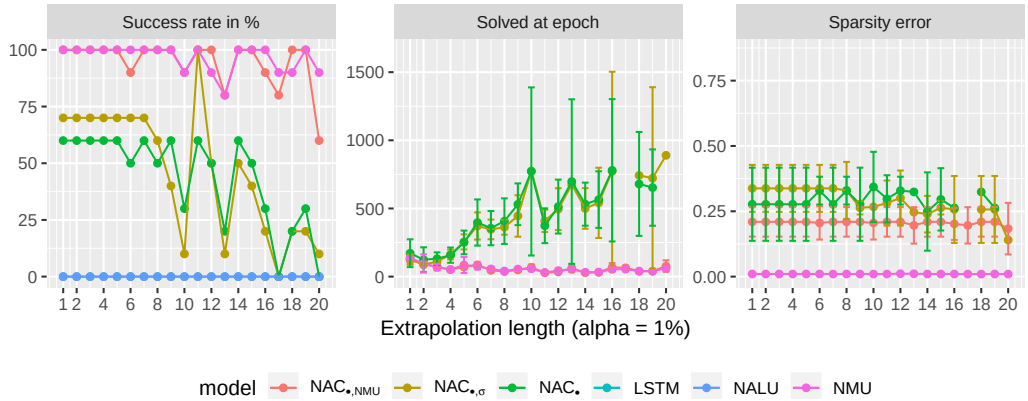


Figure 12: Shows the ability of each model to backpropergation and extrapolate to larger sequence lengths. The NMU and $\mathrm{NAC}_{\bullet,\mathrm{NMU}}$ does not use the R_z regularizer.