A ADDITIONAL EXPERIMENTS

Additional experiments were run to validate the performance of the specialized activation functions discovered by PANGAEA.

A.1 SCALING UP

PANGAEA discovered specialized activation functions for WRN-10-4, ResNet-v1-56, and ResNet-v2-56. Table 3 shows the performance of these activation functions when paired with the larger WRN-16-8, ResNet-v1-110, and ResNet-v2-110 architectures. Due to time constraints, ReLU is the only baseline activation function in these experiments.

Two of the three functions discovered for WRN-10-4 outperform ReLU with WRN-16-8, and all three functions discovered for ResNet-v2-56 outperform ReLU with ResNet-v2-110. Surprisingly, ReLU achieves the highest accuracy for ResNet-v1-110. Since activation functions are part of the skip connections in ResNet-v1, it is more difficult to achieve high performance with custom activation functions when the architecture is very deep. Activation functions are not part of the skip connections in ResNet-v2, making it easier to achieve high performance with specialized activation functions on this very deep architecture.

Evolving novel activation functions can be computationally expensive. The results in Table 3 suggest that in future work it may be possible to alleviate this cost by evolving activation functions for smaller architectures, and then using the discovered functions with larger architectures.

A.2 ADJUSTING ARCHITECTURE WIDTH AND DEPTH

To further investigate the effect of network size on the performance of novel activation functions, two specialized activation functions were paired with neural networks of different widths and depths. Due to time constraints, the results in this experiment are based on single training runs.

Wide Residual Networks The specialized activation function $\log(\sigma(\alpha x)) \cdot \beta \text{arcsinh}(x)$ was discovered for a Wide ResNet of depth 10 and width four (WRN-10-4). Figure 6 shows the performance of this function when paired with Wide ResNets of different depths and widths.

For all widths tested, $\log(\sigma(\alpha x)) \cdot \beta \text{arcsinh}(x)$ outperforms ReLU, albeit with diminishing returns as the width becomes large. This result implies that $\log(\sigma(\alpha x)) \cdot \beta \text{arcsinh}(x)$ gives the network more representational power than ReLU. As the width of the architecture is increased, the additional network parameters partially offset this advantage, explaining the decreasing relative improvement of $\log(\sigma(\alpha x)) \cdot \beta \text{arcsinh}(x)$ over ReLU.

For a fixed architecture width of four, $\log(\sigma(\alpha x)) \cdot \beta \text{arcsinh}(x)$ outperforms ReLU only when the depth is 10 and 16. Surprisingly, as the depth is increased to 22 and beyond, the performance of $\log(\sigma(\alpha x)) \cdot \beta \text{arcsinh}(x)$ drops. This result suggests that $\log(\sigma(\alpha x)) \cdot \beta \text{arcsinh}(x)$ is specialized to shallow architectures.

Preactivation Residual Networks The specialized activation function Softplus(ELU(x)) was discovered for a Preactivation ResNet of depth 56 (ResNet-v2-56). Figure 6 shows the performance of this function when paired with Preactivation ResNets of different depths. Unlike with the Wide ResNets, there is no clear increase

Table 3: Specialized activation functions discovered for WRN-10-4, ResNet-v1-56, and ResNet-v2-56 are evaluated on larger versions of those architectures: WRN-16-8, ResNet-v1-110, and ResNet-v2-110, respectively. CIFAR-100 test accuracy is reported as the median of three runs, with mean ± sample standard deviation in parenthesis. Specialized activation functions successfully transfer to WRN-16-8 and ResNet-v2-110, outperforming ReLU.

WRN-16-8	
$\log(\sigma(\alpha x)) \cdot \text{arcsinh}(x)$	**78.42** (78.34 ± 0.20)
$\log(\sigma(\alpha x)) \cdot \beta\text{arcsinh}(x)$	78.38 (78.36 ± 0.17)
$-\text{Swish}(\text{Swish}(x))$	77.90 (78.00 ± 0.35)
ReLU	78.14 (78.15 ± 0.03)
ResNet-v1-110	
$\alpha x - \beta \log(\sigma(\gamma x))$	70.88 (70.85 ± 0.50)
$\alpha x - \log(\sigma(\beta x))$	70.40 (70.34 ± 0.60)
$\max\{\text{Swish}(x), 0\}$	70.30 (70.36 ± 0.56)
ReLU	**71.15** (71.23 ± 0.25)
ResNet-v2-110	
Softplus(ELU(x))	**77.34** (77.14 ± 0.38)
$\min\{\log(\sigma(x)), \alpha \log(\sigma(\beta x))\}$	76.99 (76.93 ± 0.19)
SELU(Swish(x))	77.04 (76.96 ± 0.14)
ReLU	76.35 (76.34 ± 0.11)

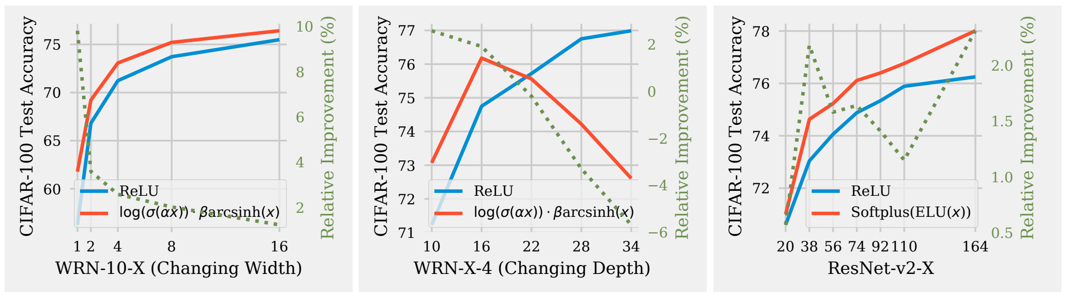

Figure 6: CIFAR-100 test accuracy for different neural networks and activation functions. Accuracy with ReLU is shown in blue, and accuracy with the specialized activation functions in red. The relative improvement of the specialized functions over ReLU is shown as a dotted green line, according to the axis values on the right of each plot. **Left:** The depth of Wide ResNet is fixed at 10, and the width varies from 1 to 16. **Center:** The depth of Wide ResNet varies from 10 to 34, while the width is fixed at four. **Right:** The depth of Preactivation ResNet ranges from 20 to 164. The width and depth of a network can affect how much a specialized activation function outperforms ReLU.

or decrease in relative improvement over ReLU as depth increases. Impressively, ResNet-v2-164 with $\text{Softplus}(\text{ELU}(x))$ achieved test set accuracy 78.01, outperforming the accuracy of ResNet-v2-1001 with ReLU (77.29) as reported by He et al. (2016b).

B Training Details

Wide Residual Network (WRN-10-4) When measuring final performance after evolution, the standard WRN setup is used; all ReLU activations in WRN-10-4 are replaced with the evolved activation function, but no other changes to the architecture are made. The network is optimized using stochastic gradient descent with Nesterov momentum 0.9. The network is trained for 200 epochs; the initial learning rate is 0.1, and it is decreased by a factor of 0.2 after epochs 60, 120, and 160. Dropout probability is set to 0.3, and L2 regularization of 0.0005 is applied to the weights. Data augmentation includes featurewise center, featurewise standard deviation normalization, horizontal flip, and random 32×32 crops of images padded with four pixels on all sides. This setup was chosen to mirror the original WRN setup (Zagoruyko & Komodakis, 2016) as closely as possible.

During evolution of activation functions, the training is compressed to save time. The network is trained for only 100 epochs; the learning rate begins at 0.1 and is decreased by a factor of 0.2 after epochs 30, 60, and 80. Empirically, the accuracy achieved by this shorter schedule is sufficient to guide evolution; the computational cost saved by halving the time required to evaluate an activation function can then be used to search for additional activation functions.

Residual Network (ResNet-v1-56) As with WRN-10-4, when measuring final performance with ResNet-v1-56, the only change to the architecture is replacing the ReLU activations with an evolved activation function. The network is optimized with stochastic gradient descent and momentum 0.9. Dropout is not used, and L2 regularization of 0.0001 is applied to the weights. In the original ResNet experiments (He et al., 2016a), an initial learning rate of 0.01 was used for 400 iterations before increasing it to 0.1, and further decreasing it by a factor of 0.1 after 32K and 48K iterations. An iteration represents a single forward and backward pass over one training batch, while an epoch consists of training over the entire training dataset. In this paper, the learning rate schedule is implemented by beginning with a learning rate of 0.01 for one epoch, increasing it to 0.1, and then decreasing it by a factor of 0.1 after epochs 91 and 137. (For example, (48K iterations / 45K training images) * batch size of 128 $\approx$ 137.) The network is trained for 200 epochs in total. Data augmentation includes a random horizontal flip and random 32×32 crops of images padded with four pixels on all sides, as in the original setup (He et al., 2016a).

When evolving activation functions for ResNet-v1-56, the learning rate schedule is again compressed. The network is trained for 100 epochs; the initial warmup learning rate of 0.01 still lasts one epoch, the learning rate increases to 0.1, and then decreases by a factor of 0.1 after epochs 46 and 68.

When evolving activation functions, their relative performance is more important than the absolute accuracies they achieve. The shorter training schedule is therefore a cost-efficient way of discovering high-performing activation functions.

Preactivation Residual Network (ResNet-v2-56) The full training setup, data augmentation, and compressed learning rate schedule used during evolution for ResNet-v2-56 are all identical to those for ResNet-v1-56 with one exception: with ResNet-v2-56, it is not necessary to warm up training with an initial learning rate of 0.01 (He et al., 2016b), so this step is skipped.

C Implementation and Compute Requirements

High-performance computing in two clusters is utilized for the experiments. One cluster uses HTCondor (Thain et al., 2005) for scheduling jobs, while the other uses the Slurm workload manager. Training is executed on GeForce GTX 1080 GPUs on both clusters. When a job begins executing, a parent activation function is selected by sampling $S = 16$ functions from the $P = 64$ most recently evaluated activation functions. This is a minor difference from the original regularized evolution (Real et al., 2019), which is based on a strict sliding window of size P. This approach may give extra influence to some activation functions, depending on how quickly or slowly jobs are executed in each of the clusters. In practice the method is highly effective; it allows evolution to progress quickly by taking advantage of extra compute when demand on the clusters is low.

It is difficult to know ahead of time how computationally expensive the evolutionary search will be. Some activation functions immediately result in an undefined loss, causing training to end. In that case only a few seconds have been spent and another activation function can immediately be evaluated. Other activation functions train successfully, but their complicated expressions result in longer-than-usual training times. In these experiments, evolution for WRN-10-4 took 2,314 GPU hours, evolution for ResNet-v1-56 took 1,594 GPU hours, and evolution for ResNet-v2-56 took 2,175 GPU hours. These numbers do not include costs for reranking and repeated runs in the final experiments. Although substantial, the computational cost is negligible compared to the cost in human labor in designing activation functions. Evolution of parametric activation functions requires minimal manual setup and delivers automatic improvements in accuracy.

D Baseline Activation Function Details

Name	Definition	Reference(s)
ReLU	$\max\{x, 0\}$	Nair & Hinton (2010)
ELiSH	$\frac{x}{1+e^{-x}}$ `if` $x \geq 0$ `else` $\frac{e^x-1}{1+e^{-x}}$	Basirat & Roth (2018)
ELU	x `if` $x \geq 0$ `else` $\alpha(e^x - 1)$, with $\alpha = 1$	Clevert et al. (2015)
GELU	$x\Phi(x)$, with $\Phi(x) = P(X \leq x), X \sim \mathcal{N}(0, 1)$, approximated as $0.5x(1 + \tanh[\sqrt{2/\pi}(x + 0.044715x^3)])$	Hendrycks & Gimpel (2016)
HardSigmoid	$\max\{0, \min\{1, 0.2x + 0.5\}\}$	
Leaky ReLU	x `if` $x \geq 0$ `else` $0.01x$	Maas et al. (2013)
Mish	$x \cdot \tanh(\text{Softplus}(x))$	Misra (2019)
SELU	λx `if` $x \geq 0$ `else` $\lambda\alpha(e^x - 1)$, with $\lambda = 1.05070098$, $\alpha = 1.67326324$	Klambauer et al. (2017)
sigmoid	$(1 + e^{-x})^{-1}$	
Softplus	$\log(e^x + 1)$	
Softsign	$x/(\lvert x\rvert + 1)$	
Swish	$x \cdot \sigma(x)$, with $\sigma(x) = (1 + e^{-x})^{-1}$	Ramachandran et al. (2018) and Elfwing et al. (2018)
tanh	$\frac{e^x - e^{-x}}{e^x + e^{-x}}$	

Table 4: Baseline activation functions from the operator search space (Table 1) and final results (Table 2).