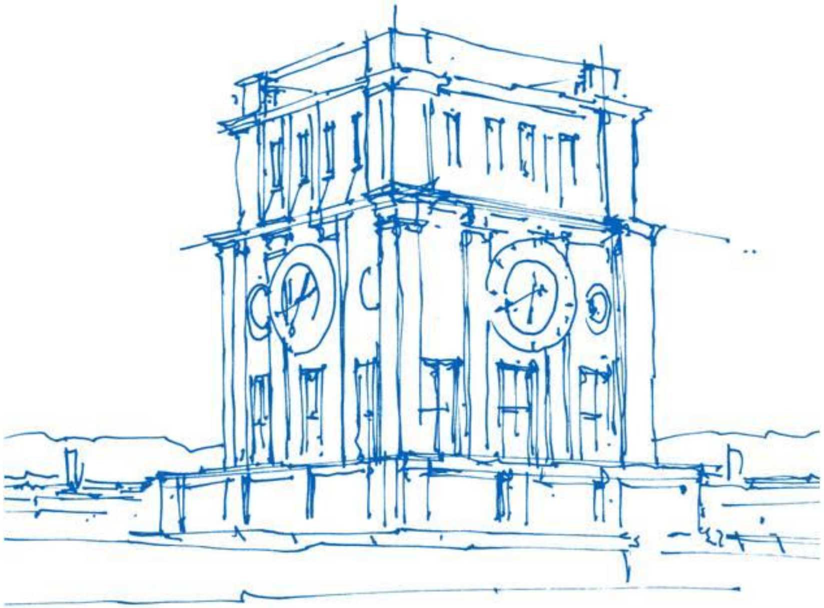
Searching for Activation Functios

Jenny Seidenschwarz

Technische Universität München

Seminar Course AutoML

Munich, 4th of July 2019



Uhrenturm der TUM

Lehrstuhl für Musterverfahren
Fakultät für Mustertechnik
Technische Universität München

Activation Functions

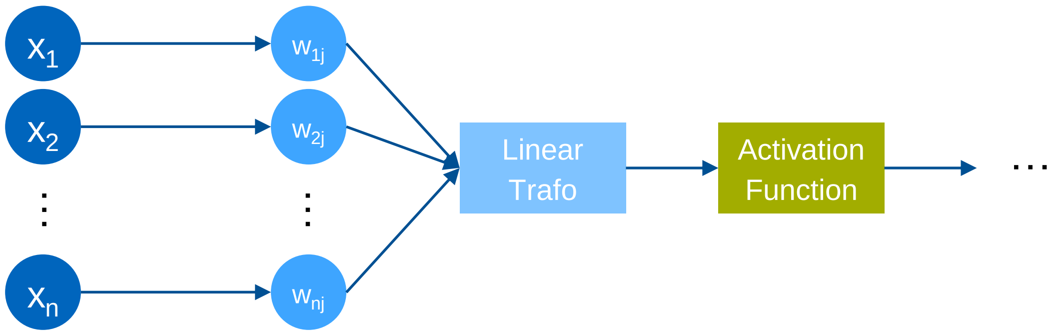


Figure: Basic structure neural network

State of the art default scalar activation function: ReLU $max(0, x)$

- Gradient preserving property
- More easy to optimize

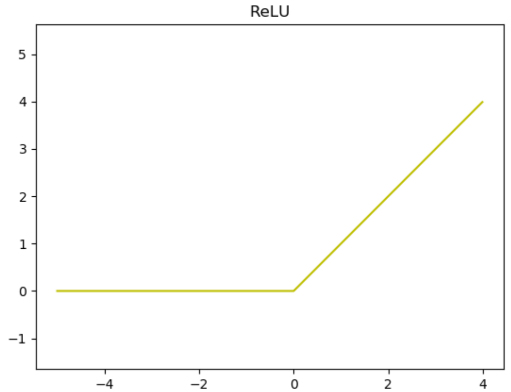


Figure: ReLU activation function

Research Goal

Find new scalar activation functions …
… using automated search technique …
… compare them systematically to existing activation functions …
… across multiple different challenging datasets!

Automated Search

Search Space

Challenge: balance size and expressivity of search space

→ Simple binary expression tree [1]

→ Selection of unary and binary functions

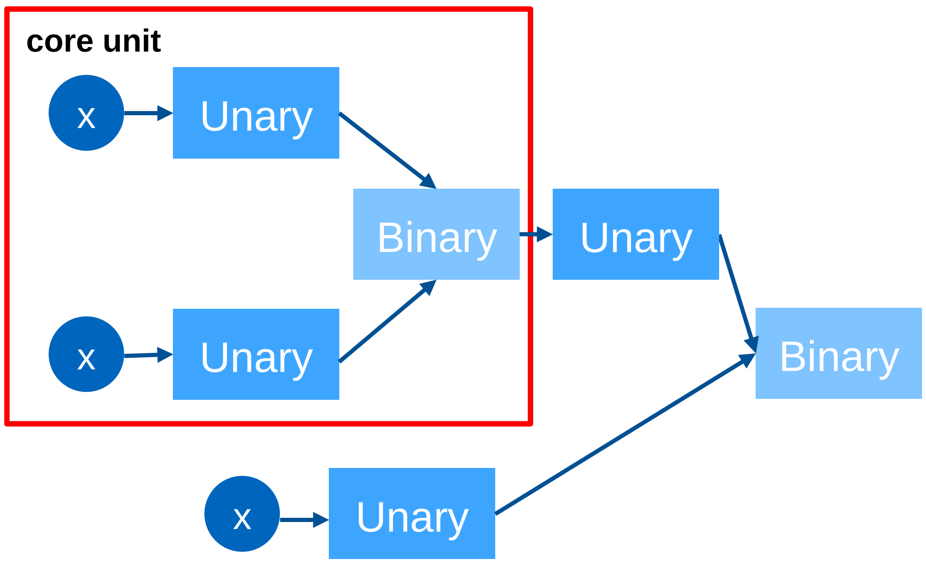
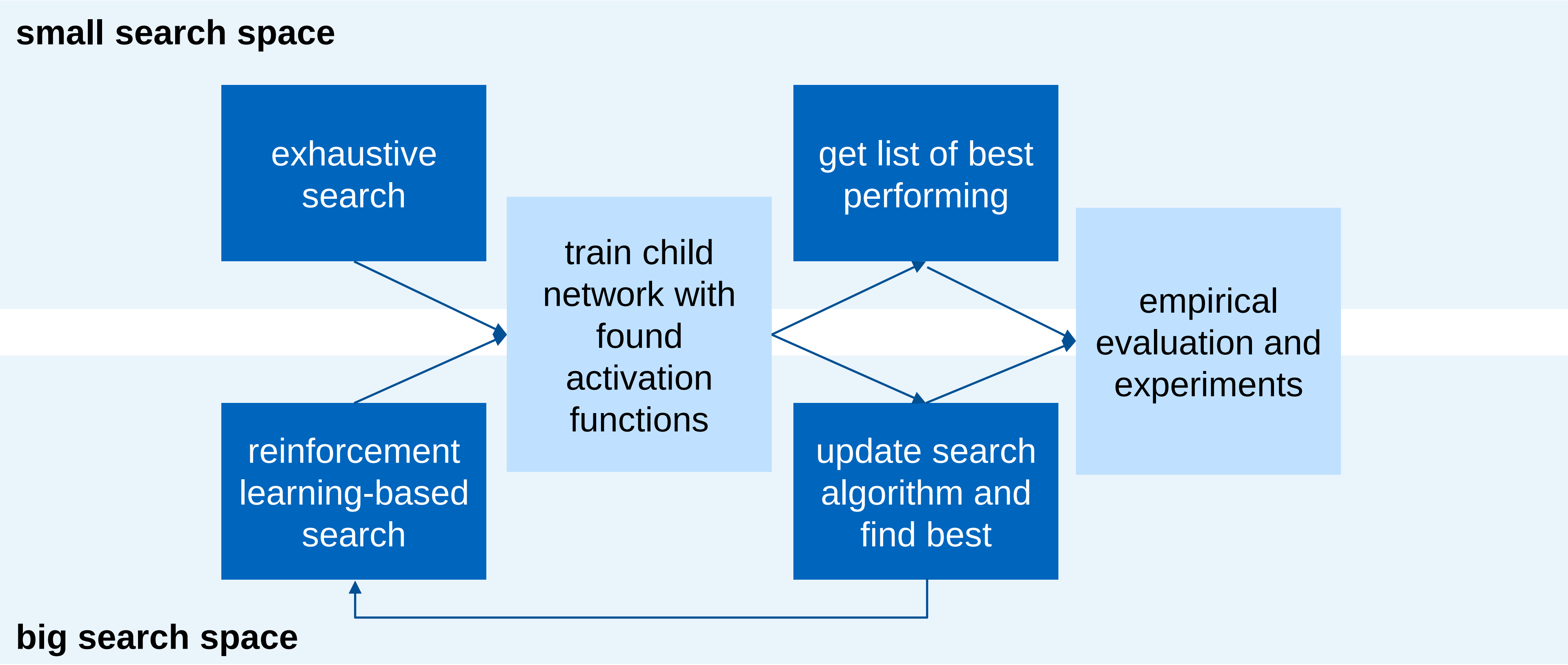


Figure: Core Unit, adapted from [5]

Unary: x, $-x$, $|x|$, x^2, x^3, $\sqrt{x}$, βx, $x+\beta$, $\log(|x|+\epsilon)$, $\exp(x)$, $\sin(x)$, $\cos(x)$, $\sinh(x)$, $\cosh(x)$, $\tanh(x)$, $tan^{-1}(x)$, $sinh^{-1}(x)$, $sinc(x)$, $\max(0,x)$, $\min(0,x)$, $\sigma(x)$, $\log(1+\exp(x))$, $\exp(-x^2)$, $\mathrm{erf}(x)$, β

BInary: x_1+x_2, $x_1 x_2$, x_1-x_2, $\frac{x_1}{x_2+\epsilon}$, $\max(x_1,x_2)$, $\min(x_1,x_2)$, $\sigma(x_1)x_2$, $\exp(-\beta(x_1-x_2)^2)$, $\exp(-\beta|x_1-x_2|)$, $\beta x_1+(1-\beta)x_2$

Search approach



RNN-Controller

RNN-controller [2] with domain specific language [1]

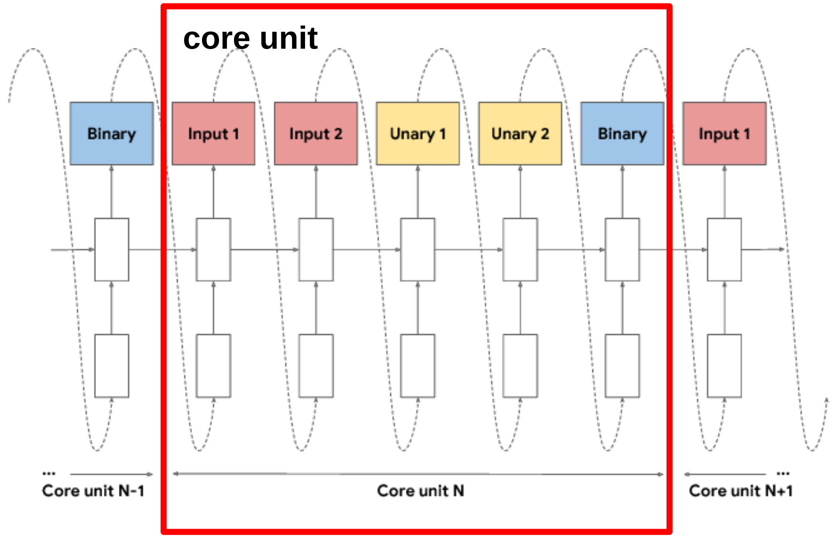


Figure: RNN-Controller architecture [5]

Train batch of generated activation functions

- ResNet-20
- Image classification on CIFAR-10
- 10k steps

RNN-Controller update

Policy gradient methods: $\pi\,(a_t|s_t, \theta_c)$ → $\Delta\theta \Leftarrow \alpha\,\nabla\mathcal{L}_{\theta c}$

RNN Controller with REINFORCE:

- Objective function $\mathcal{L}_{\theta c} = \mathbb{E}_{\pi_{\theta,\tau}}[G_t]$, where $G_t = \sum_{k=t}^{T-1} \gamma^{k-t} r_{k+1}$

RNN Controller with PPO:

→ Clipping ensures updates in „trust region“

→ Sample efficient

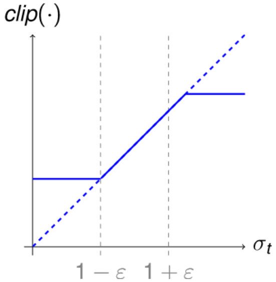


Figure: clipping function, adapted from [3]

- Objective function

$$LCLIP(\theta) = \mathbb{E}_t[\min\{\sigma_t G_t, clip(\sigma_t, 1-\ \varepsilon, 1+\ \varepsilon)G_t\}], \qquad with\ \ \sigma_t = \frac{\pi_\theta(a_t \mid s_t)}{\pi_{\theta_{old}}(a_t \mid s_t)}$$

RNN-Controller update

Policy gradient methods: $\pi\,(a_t|s_t, \theta_c)$ → $\Delta\theta \Leftarrow \alpha\,\nabla\mathcal{L}_{\theta c}$

RNN Controller with REINFORCE:

- Objective gradient — One child network: $\nabla\mathcal{L}_{\theta c} = \sum_{t=1}^{T} \nabla_{\theta c}\, log\, \pi\,(a_t|s_t, \theta_c)\,(G - b)$

RNN Controller with PPO:

- Objective gradient

$$\nabla\mathcal{L}_{\theta_c} = \begin{cases} \frac{1}{m}\sum_{k=1}^{m}\sum_{t=1}^{T}\nabla_{\theta_c} log\sigma_t(G_k - b) & \sigma_t G_k \leq clip \\ 0 & \sigma_t G_t > clip \end{cases}$$

→ G_k = accuracy of child network
→ b = exponential moving average of rewards

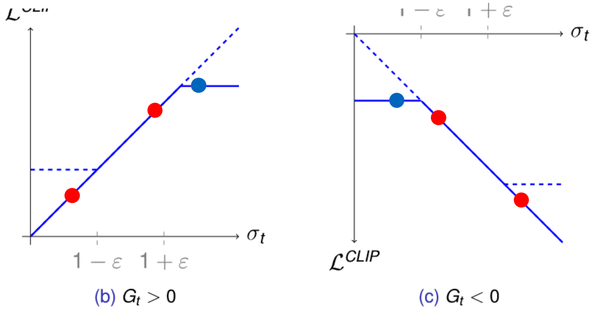


Figure: clipping function, adapted from [3]

Findings on Activation Functions

Findings on Activation Functions

1. 1-2 core units perform best
2. Top activation functions always take raw preactivation x as input to final binary function
3. Periodic functions (sin, cos, etc.) used by some top performing activation functions
4. Activation functions that use division perform poorly

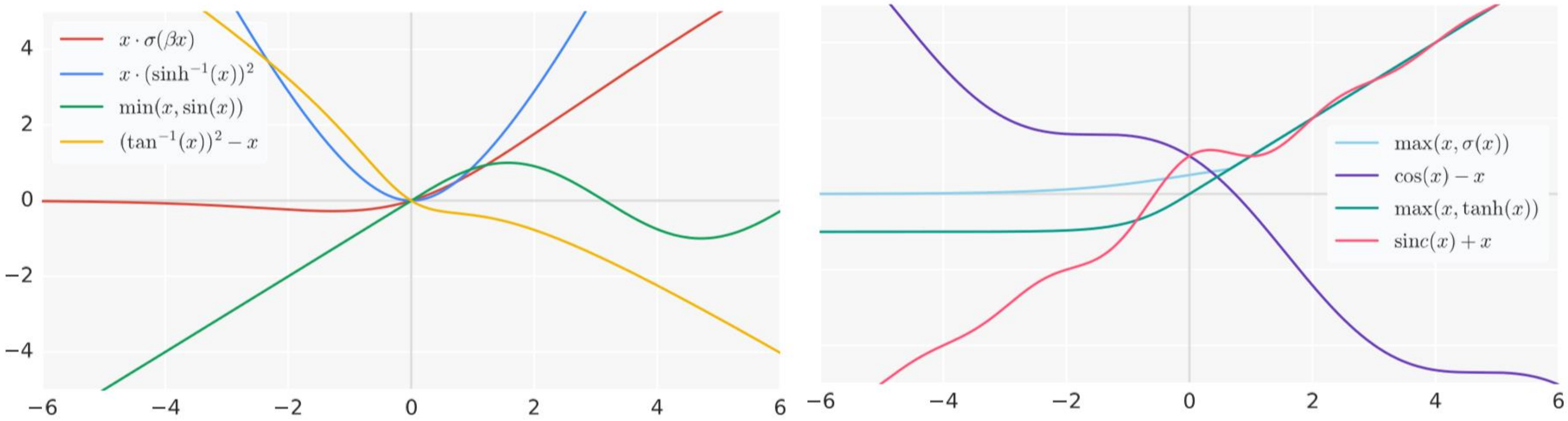


Figure: best 8 activation functions [5]

Validation of Performance

Experiments to ensure generalization to deeper networks

Swish

Function	RN	WRN	DN
ReLU $[\max(x, 0)]$	93.8	95.3	94.8
$x \cdot \sigma(\beta x)$	94.5	95.5	94.9
$\max(x, \sigma(x))$	94.3	95.3	94.8
$\cos(x) - x$	94.1	94.8	94.6
$\min(x, \sin(x))$	94.0	95.1	94.4
$(\tan^{-1}(x))^2 - x$	93.9	94.7	94.9
$\max(x, \tanh(x))$	93.9	94.2	94.5
$\text{sinc}(x) + x$	91.5	92.1	92.0
$x \cdot (\sinh^{-1}(x))^2$	85.1	92.1	91.1

(a) CIFAR-10 accuracy

Function	RN	WRN	DN
ReLU $[\max(x, 0)]$	74.2	77.8	83.7
$x \cdot \sigma(\beta x)$	75.1	78.0	83.9
$\max(x, \sigma(x))$	74.8	78.6	84.2
$\cos(x) - x$	75.2	76.6	81.8
$\min(x, \sin(x))$	73.4	77.1	74.3
$(\tan^{-1}(x))^2 - x$	75.2	76.7	83.1
$\max(x, \tanh(x))$	74.8	76.0	78.6
$\text{sinc}(x) + x$	66.1	68.3	67.9
$x \cdot (\sinh^{-1}(x))^2$	52.8	70.6	68.1

(b) CIFAR-100 accuracy

Figure: Generalization to deeper architectures of 8 best activation functions found [5]

Swish

$$f(x) = x\,\sigma(\beta x)$$

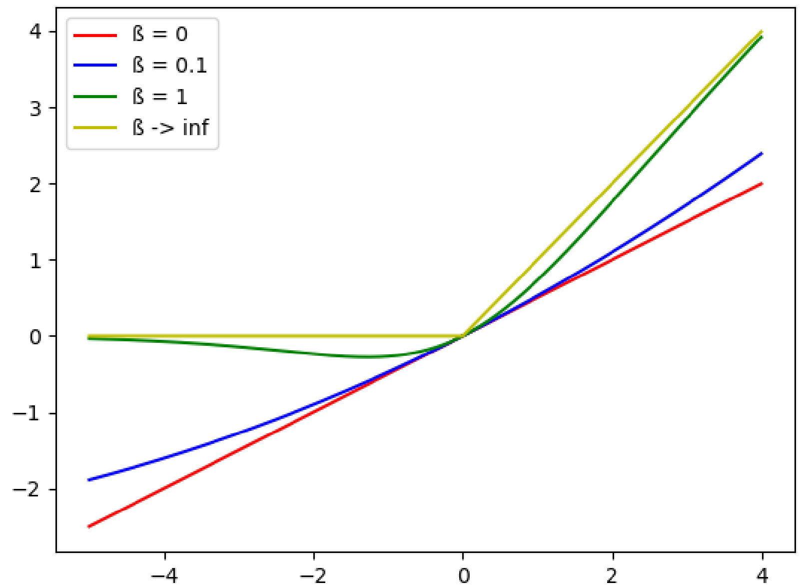


Figure: Swish activation function for different β values

- Nonlinearly interpolation between ReLU and linear function
- Smooth function
- Non-monotinoc function
- Unbounded above and bounded below (like ReLU)

Benchmark of Swish

Further Experiments with Swish

Benchmarked Swish to ReLU and other baseline activation functions

ReLU $f(x) = \begin{cases} x, if\ x \geq 0 \\ 0, if\ x < 0 \end{cases}$

Leaky ReLU $f(x) = \begin{cases} x, if\ x \geq 0 \\ \alpha x, if\ x < 0 \end{cases}$, where $\alpha = 0.01$ is learnable

Parametric ReLU $f(x) = \begin{cases} x, if\ x \geq 0 \\ \alpha x, if\ x < 0 \end{cases}$, where α is learnable

Softplus $f(x) = \log(1 + \exp(1))$

ELU $f(x) = \begin{cases} x, if\ x \geq 0 \\ \alpha(\exp(x) - 1), if\ x < 0 \end{cases}$, where α = 1.0

Scaled ELU $f(x) = \lambda \begin{cases} x, if\ x \geq 0 \\ \alpha(\exp(x) - 1), if\ x < 0 \end{cases}$, where α = 1.6733, λ = 1.0507

GELU $f(x) = x\,\Phi(x)$, where $\Phi(x)$ = cumulative dist

- Different models
- Different challenging real world datasets
- Test with fixed β = 1 and trainable β

Further Experiments with Swish

CIFAR 10 and 100: ResNet-164, Wide ResNet 28-10 and DenseNet 100-12

- Median of 5 runs for comparison

ImageNet: Inception-ResNet-v2, Inception-v4, Inception-v3, MobileNet and Mobile NASNet-A

- Fixed number of steps, 3 learning rates with RMSProp
- Epsecially good performance on mobile sized modelm slightly underperform Inception-v4

English-German-translation: 12 layer Base Transformer

- Two different learning rates, 300K steps with Adam optimizer

Baselines	**ReLU**	**LReLU**	**PReLU**	**Softplus**	**ELU**	**SELU**	**GELU**
Swish $>$ Baseline	9	7	6	6	8	8	8
Swish $=$ Baseline	0	1	3	2	0	1	1
Swish $<$ Baseline	0	1	0	1	1	0	0

Figure: Overview performace in experiments [5]

Performance of Swish

Swish – learnable parameter β

Learnable β:

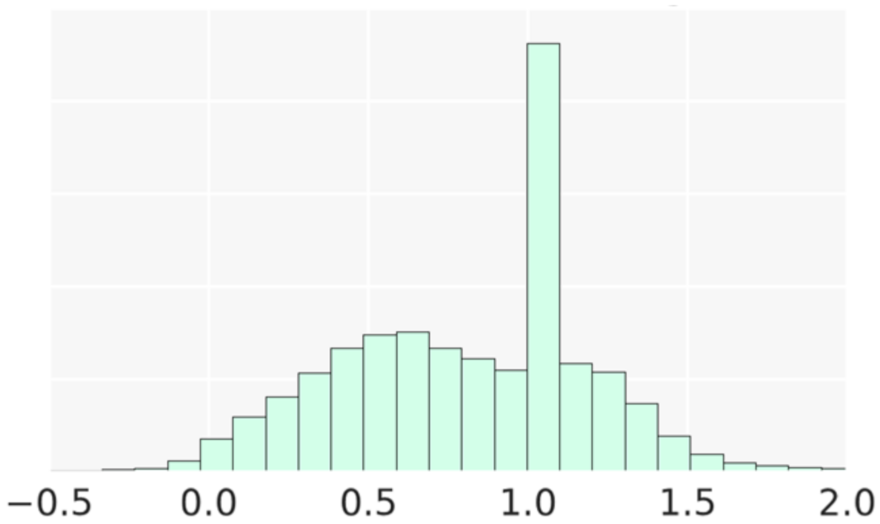


Figure: distribution of trained β on Mobile NASNet-A [5]

Swich – Challenging current Belief

Non-monotinic bump for $x < 0$

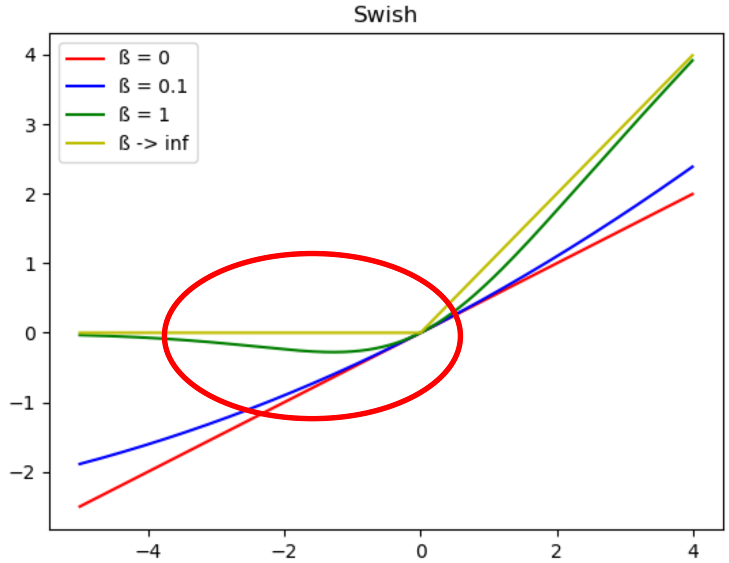


Figure: Swish function for different β values with non-monotonic bump

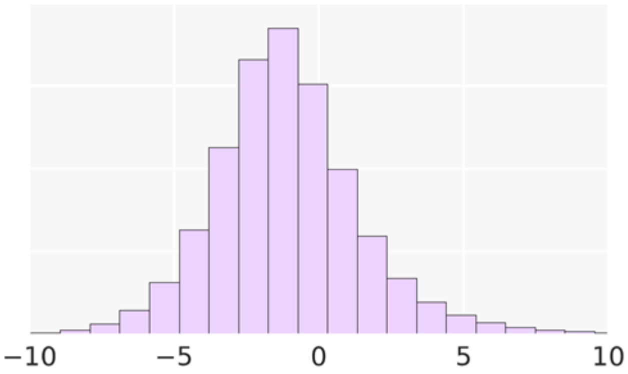


Figure: Preactivations for $\beta = 1$ on ResNet-32 [5]

Swich – Challenging current Belief

No gradient preserving characteristics of derivative:

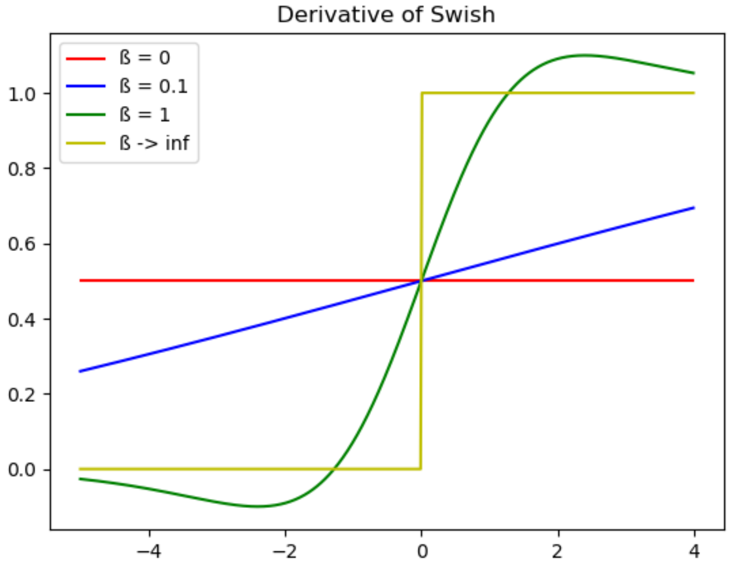


Figure: Derivative of swish function for different β values

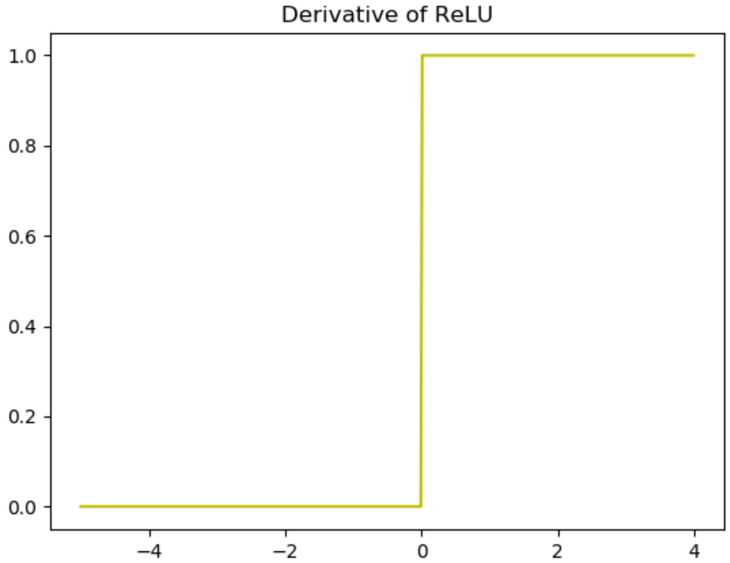


Figure: Derivative of ReLU

Conclusion

Main contributions:

- Used a search space as in [1] to find activation functions with a RNN controller [2], that was updated with PPO [3]
- Systematically compared activation functions
- Found new activation function that constantly outperform or is on par with ReLU

Critical aspects:

- Search space restricts results
- Search space designed after human intuition
- Restriction of training steps and training on small architectures might suppress even better activation functions

Future research:

- Only two core units, but more unary and binary functions
- Also take non-scalar activation functions into account

References

[1] Bello, I. & Zoph, B. & Vasudevan, V. & Le, Quoc V. (2017). Neural Optimizer Search with Reinforcement Learning.
[2] Zoph, B., & Le, Quoc V. (2017). Neural Architecture Search with Reinforcement Learning. ArXiv, abs/1611.01578.
[3] Schulman, J., Wolski, F., Dhariwal, P., Radford, A., & Klimov, O. (2017). Proximal Policy Optimization Algorithms. *ArXiv, abs/1707.06347*.
[4] Elfwing, S. & Uchibe, E. & Doya, K. (2018). Sigmoid-Weighted Linear Units for Neural Network Function Approximation in Reinforcement Learning. Neural Networks. 107. 10.1016/j.neunet.2017.12.012.
[5] Ramachandran, P., Zoph, B., & Le, Q.V. (2018). Searching for Activation Functions. ArXiv, abs/1710.05941.

Back-up

Things to note

If you want to use Swish:

- Already implemented in tensorflow as tf.nn.swish(x)
- When using batch norm: set scale parameter
- Derivative of swish: $f'(x) = \beta f(x) + \sigma(\beta x)(1 - \beta f(x))$

Experiment Results - CIFAR

Model	ResNet	WRN	DenseNet
LReLU	94.2	95.6	94.7
PReLU	94.1	95.1	94.5
Softplus	94.6	94.9	94.7
ELU	94.1	94.1	94.4
SELU	93.0	93.2	93.9
GELU	94.3	95.5	94.8
ReLU	93.8	95.3	94.8
Swish-1	94.7	95.5	94.8
Swish	94.5	95.5	94.8

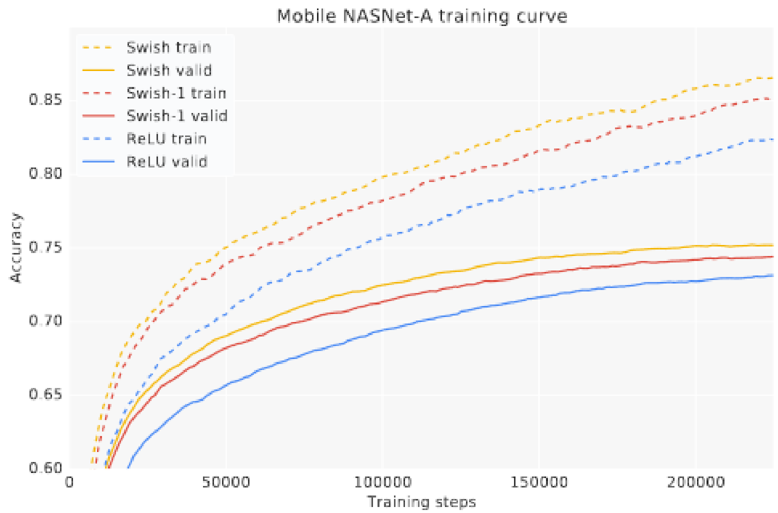
(a) CIFAR-10 accuracy

Model	ResNet	WRN	DenseNet
LReLU	74.2	78.0	83.3
PReLU	74.5	77.3	81.5
Softplus	76.0	78.4	83.7
ELU	75.0	76.0	80.6
SELU	73.2	74.3	80.8
GELU	74.7	78.0	83.8
ReLU	74.2	77.8	83.7
Swish-1	75.1	78.5	83.8
Swish	75.1	78.0	83.9

(b) CIFAR-100 accuracy

Figure: Benchmark experiments of Swish function to baseline functions on CIFAR [5]

Experiments on ImageNet



Model	Top-1 Acc. (%)			Top-5 Acc. (%)		
LReLU	73.8	73.9	74.2	91.6	91.9	91.9
PReLU	74.6	74.7	74.7	92.4	92.3	92.3
Softplus	74.0	74.2	74.2	91.6	91.8	91.9
ELU	74.1	74.2	74.2	91.8	91.8	91.8
SELU	73.6	73.7	73.7	91.6	91.7	91.7
GELU	74.6	-	-	92.0	-	-
ReLU	73.5	73.6	73.8	91.4	91.5	91.6
Swish-1	74.6	74.7	74.7	92.1	92.0	92.0
Swish	74.9	74.9	75.2	92.3	92.4	92.4

(a) Training curves of Mobile NASNet-Aon ImageNet. Best viewed in color

(b) Mobile NASNet-A on ImageNet, with3 different runs ordered by top-1 accuracy. Theadditional 2 GELU experiments are still trainingat the time of submission.

Figure: Benchmark experiments of Swish function to baseline functions on ImageNet [5]

Experiments on ImageNet

Model	Top-1 Acc. (%)			Top-5 Acc. (%)		
LReLU	79.5	79.5	79.6	94.7	94.7	94.7
PReLU	79.7	79.8	80.1	94.8	94.9	94.9
Softplus	80.1	80.2	80.4	95.2	95.2	95.3
ELU	75.8	79.9	80.0	92.6	95.0	95.1
SELU	79.0	79.2	79.2	94.5	94.4	94.5
GELU	79.6	79.6	79.9	94.8	94.8	94.9
ReLU	79.5	79.6	79.8	94.8	94.8	94.8
Swish-1	80.2	80.3	80.4	95.1	95.2	95.2
Swish	80.2	80.2	80.3	95.0	95.2	95.0

(a) Inception-ResNet-v2 on ImageNetwith 3 different runs. Note that the ELUsometimes has instabilities at the start oftraining, which accounts for the first result

Model	Top-1 Acc. (%)	Top-5 Acc. (%)
LReLU	72.5	91.0
PReLU	74.2	91.9
Softplus	73.6	91.6
ELU	73.9	91.3
SELU	73.2	91.0
GELU	73.5	91.4
ReLU	72.0	90.8
Swish-1	74.2	91.6
Swish	74.2	91.7

(b) MobileNet on ImageNet.

Figure: Benchmark experiments of Swish function to baseline functions on ImageNet [5]

Experiments on ImageNet

Model	Top-1 Acc. (%)	Top-5 Acc. (%)
LReLU	78.4	94.1
PReLU	77.7	93.5
Softplus	78.7	94.4
ELU	77.9	93.7
SELU	76.7	92.8
GELU	77.7	93.9
ReLU	78.4	94.2
Swish-1	78.7	94.2
Swish	78.7	94.0

(a) Inception-v3 on ImageNet

Model	Top-1 Acc. (%)	Top-5 Acc. (%)
LReLU	79.3	94.7
PReLU	79.3	94.4
Softplus	79.6	94.8
ELU	79.5	94.5
SELU	78.3	94.5
GELU	79.0	94.6
ReLU	79.2	94.6
Swish-1	79.3	94.7
Swish	79.3	94.6

(b) Inception-v4 on ImageNet

Figure: Benchmark experiments of Swish function to baseline functions on ImageNet [5]

Experiments on Machine Translation

Model	newstest2013	newstest2014	newstest2015	newstest2016
LReLU	26.2	27.9	29.8	33.4
PReLU	26.3	27.7	29.7	33.1
Softplus	23.4	23.6	25.8	29.2
ELU	24.6	25.1	27.7	32.5
SELU	23.7	23.5	25.9	30.5
GELU	25.9	27.3	29.5	33.1
ReLU	26.1	27.8	29.8	33.3
Swish-1	26.2	28.0	30.1	34.0
Swish	26.5	27.6	30.0	33.1

Figure: Benchmark experiments of Swish function to baseline functions on WTM English→German (BLEU score) [5]