# TAct: Optimal search through activation function space

*LXAI Workshop at NIPS 2018 Submission*

Mario Banuelos[1], Heyley Gatewood[2], Samuel Hood[3], Jonathan Scott[4] and David Uminsky[5]

[1]Department of Mathematics, California State University, Fresno
[2]Department of Mathematics and Computer Science, Stetson University
[3]Department of Mathematics, Morehouse College
[4]Department of Mathematics, Statistics, and Computer Science, Macalester College
[5]Department of Mathematics, University of San Francisco

September 2018

Since activation functions allow deep learning models to approximate functions of complex, nonlinear data, an important and sometimes simplified part of optimizing performance in a deep learning model is the choice of an activation function. Many recent deep learning architectures implement the Rectified Linear Unit, ReLU, (i.e., $\max(0, x)$) as a default choice [1, 2, 3]. Classic activation functions such as Tanh and sigmoid as well as new variations of ReLU and other functions, have provided a growing list of options for users [4, 5, 6, 7]. The traditional approach to finding the most accurate activation function is to fix a model and exhaustively enumerate the results across a finite list of functions. Another recent approach [8] using reinforcement learning lead to the discovery new activation functions. For at least some architectures, it also is not clear that the exact form of the function is crucial [9].

We study the importance of choosing an activation function; however, instead of proposing a new activation function, we study the relationship between the most commonly used ones. We propose a two-parameter, trainable Tanh activation function which we call TAct. TAct is a family of activation functions which exactly contains classic functions such as Tanh, Sigmoid and more recently Swish. In addition there are areas of the parameter space that approximate functions like ReLu arbitrarily closely. interpolates between previous and recent activation functions [8]. Although some work has been done for using series approximations for activation functions [10], we propose TAct as a framework to explore existing nonlinear transformations.

To create this parameter space, we begin by representing each of the following functions as a point in 2D space: Sigmoid (0,0), Shifted Tanh (0,1), and Swish (1,0), which form a convex hull of interpolations between these three activation functions. Our implementation initializes the value of the two parameters, $\mu$ and $\gamma$, at random from a uniform distribution on the interval [-1,1]. To create this parameter space, we define it TAct follows (see Figure 1):

$$\text{TAct}(x) := \left(\frac{\mu+1}{6}x + \frac{2-\mu}{6}\right)\left(\tanh\left(\frac{\gamma+4}{6}x\right)+1\right). \tag{1}$$

ReLU is approximated in eq:tact by considering $\mu = 2$, $\text{TAct}(x)$ approaches ReLU as $\gamma \to \infty$, though in practice $\gamma$ need not be too big, see figure 1.

Although we initialize our trainable parameters in this region, $\mu, \gamma \in \mathbb{R}$, the key to our approach is that the model will naturally select the best member of this family and yield new optimized activation functions.

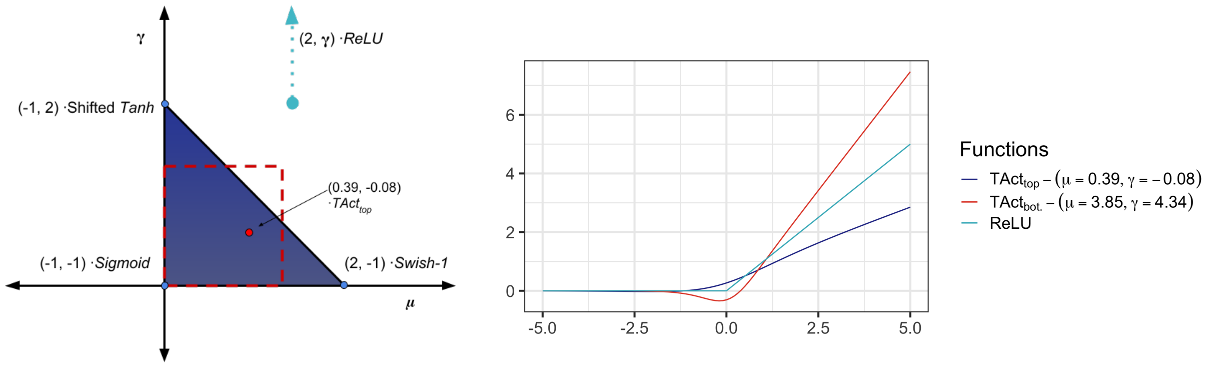

Figure 1: *Left.* Two-parameter ($\mu$-$\gamma$) TAct function parameter space along with interpolated activation functions (Shifted Tanh, Sigmoid, and Swish) by our proposed framework. Both $\mu$ and $\gamma$ are initially sampled from a [-1,1] uniform distribution, represented by the red dotted square above. When $\mu = 2$ and $\gamma$ increases, TAct approaches ReLU. *Right.* Plot of ReLU and TAct activation functions, where $\text{TAct}_{\text{top}}$ corresponds to the first convolutional layer in our Darknet implementation and $\text{TAct}_{\text{bot.}}$ corresponds to the last convolutional layer after 50 epochs.

We designed and carried out experiments to measure the effectiveness of Eq.(1) and other activation functions in multiple architectures. After applying random flipping, padding, and cropping data augmentation techniques on the CIFAR-10 dataset, we implemented a variation of the Darknet architecture with (1,2,4,6,3) convolutional blocks [11]. Instead of fixing the learning rate for each of the activation functions, we optimize learning rates as described in [12]. Figure 2 illustrates an example of the improved performance of TAct over activation functions LeakyReLU (0.1 negative slope) and ReLU. By letting the data drive the choice of activation functions, particularly in low-epoch regimes, we achieve competitive and improved test error rates when compared to other popular activation functions.

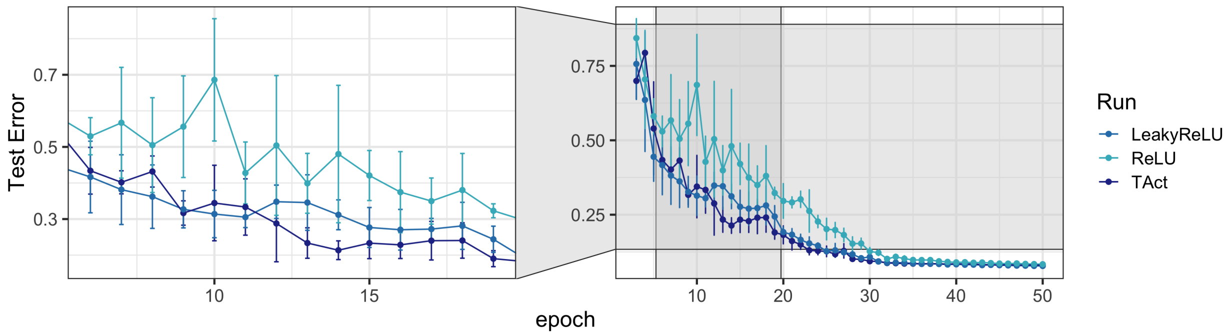

Figure 2: CIFAR-10 test error of the mean of 5 runs ($\pm$ one standard deviation) on a modified Darknet architecture with 1, 2, 4, 6, and 3 convolutional blocks for the activation functions LeakyReLU (0.1 negative slope), ReLU, and TAct. We observe an improved test error within less epochs compared using our proposed method.

## References

[1] A. Krizhevsky and G. Hinton. *Learning multiple layers of features from tiny images.* Tech. rep. Citeseer, 2009.

[2] K. Jarrett, K. Kavukcuoglu, Y. LeCun, et al. "What is the best multi-stage architecture for object recognition?" In: *Computer Vision, 2009 IEEE 12th International Conference on.* IEEE. 2009, pp. 2146–2153.

[3] V. Nair and G. E. Hinton. "Rectified linear units improve restricted boltzmann machines". In: *Proceedings of the 27th international conference on machine learning (ICML-10).* 2010, pp. 807–814.

[4] A. L. Maas, A. Y. Hannun, and A. Y. Ng. "Rectifier nonlinearities improve neural network acoustic models". In: *Proc. icml.* Vol. 30. 1. 2013, p. 3.

[5] K. He et al. "Delving deep into rectifiers: Surpassing human-level performance on imagenet classification". In: *Proceedings of the IEEE international conference on computer vision.* 2015, pp. 1026–1034.

[6] D.-A. Clevert, T. Unterthiner, and S. Hochreiter. "Fast and accurate deep network learning by exponential linear units (elus)". In: *arXiv preprint arXiv:1511.07289* (2015).

[7] G. Klambauer et al. "Self-normalizing neural networks". In: *Advances in Neural Information Processing Systems.* 2017, pp. 971–980.

[8] P. Ramachandran, B. Zoph, and Q. V. Le. "Searching for Activation Functions". In: *CoRR* abs/1710.05941 (2017). arXiv: `1710.05941`. URL: `http://arxiv.org/abs/1710.05941`.

[9] I. Goodfellow et al. *Deep learning.* Vol. 1. MIT press Cambridge, 2016.

[10] H. Chung, S. J. Lee, and J. G. Park. "Deep neural network using trainable activation functions". In: *Neural Networks (IJCNN), 2016 International Joint Conference on.* IEEE. 2016, pp. 348–352.

[11] J. Redmon and A. Farhadi. "Yolov3: An incremental improvement". In: *arXiv preprint arXiv:1804.02767* (2018).

[12] L. N. Smith. "Cyclical learning rates for training neural networks". In: *Applications of Computer Vision (WACV), 2017 IEEE Winter Conference on.* IEEE. 2017, pp. 464–472.