# ADACT: LEARNING TO OPTIMIZE ACTIVATION FUNCTION CHOICE THROUGH ADAPTIVE ACTIVATION MODULES

**Ritabrata Maiti**
Nanyang Technological University
50 Nanyang Ave, Singapore 639798
ritabrat001@e.ntu.edu.sg

## ABSTRACT

This paper presents an innovative approach to enhancing neural network performance through the development and implementation of an adaptive activation function, termed adaptive activation (AdAct). AdAct amalgamates various well-established and novel activation functions into a single, learnable framework, allowing dynamic adaptation to specific network layers' needs. We explore the effectiveness of ReLU and its variants, including ELU, LReLU, PReLU, RReLU, and more recent functions like Swish and Mish, integrating them into the AdAct function. Employing ConvNet variants across FMNIST, CIFAR10, SVHN and FER datasets, our study empirically assesses each function's contribution and demonstrates AdAct's potential in optimizing neural networks, especially in selecting optimal activation functions for diverse tasks.

## 1 INTRODUCTION

In the domain of deep learning, the efficacy of neural networks is heavily influenced by the choice of activation functions. Historically, the Rectified Linear Unit (ReLU) has been the cornerstone activation function Xu et al. (2015), renowned for its simplicity and effectiveness in various network architectures (Ramachandran et al., 2017). Despite the emergence of numerous alternatives, consistent outperformance of ReLU has remained elusive. Recent developments, however, have seen a shift towards exploring innovative activation functions discovered via automatic search techniques. This led to the discovery of functions like Swish, which demonstrated improved performance in deeper models and complex datasets (Ramachandran et al., 2017).

Further advancements introduced adaptive activation functions, such as ACON, which not only enhanced model performance but also provided a broader design space, making functions like Swish a subset of these new families (Ma et al., 2021b). This progression underscores a growing need to reevaluate traditional activation functions and explore customizable, dynamic alternatives. However, there is a noted gap between the activation functions commonly used in practice versus those emerging from recent research, suggesting a disconnect in the application of these novel functions in practical scenarios (Nwankpa et al., 2018).

Addressing this gap, our research introduces the adaptive activation (AdAct) function, a method that combines a variety of activation functions, both classic and contemporary, into a single learnable framework. This approach allows for the empirical assessment and dynamic adaptation of activation functions, tailored to the specific needs of different layers within a neural network. We implement this novel AdAct function in several convolutional neural network (ConvNet) variants, assessing their performance across the FMNIST, CIFAR10, FER, and SVHN datasets. Our study aims to demonstrate the potential of adaptive activation functions in enhancing neural network performance, providing a solution to the challenge of selecting the optimal combination of activation functions for diverse deep learning tasks.

## 2 METHODS

In our exploration of activation functions, we examine the Rectified Linear Unit (ReLU) and its variants, including, Exponential Linear Unit (ELU), Leaky ReLU (LReLU), Parametric ReLU (PReLU), and Randomized ReLU (RReLU) (Xu et al., 2015), as well as other notable functions like Swish, or Sigmoid Linear Unit (SiLU) which was discovered via neural architecture search (Zoph & Le, 2017) and Mish, an activation function that further builds on the Swish (Misra, 2020). In recent years, ReLU and its variants have become the predominant activation functions in use (Xu et al., 2015), while the more recently discovered Swish and Mish demonstrate good performance and great potential (Ma et al., 2021a).

The core of our method is the AdAct module which applies a weighted sum of the aforementioned activation functions. We define a vector of weights $\mathbf{w} \in \mathbb{R}^7$ as learnable parameters within a neural network model, that are optimized end-to-end by backpropagation. These weights are then scaled using the softmax function as a convenient way to scale the weights to be positive values and within a finite range. The output of the activation function is then the sum of each activation function applied to the input $x$, weighted by the softmax-scaled values of $\mathbf{w}$. Formally, the output for a given input $x$ is given by:

$$\text{AdAct}(x) = \sum_{i=1}^{7} \text{softmax}(\mathbf{w})_i \cdot \text{Act}_i(x)$$

where $\text{Act}_i$ represents one of the seven activation functions under consideration, i.e. $\text{Act}_i \in \{\text{ReLU, ELU, LReLU, PReLU, RReLU, SiLU, Mish}\}$ and the softmax function is defined as $\text{softmax}(\mathbf{z})_i = \frac{e^{z_i}}{\sum_{j=1}^{7} e^{z_j}}$. This method enables empirical evaluation of each activation function's impact and introduces a learnable activation unit that could eliminate the need for fixed activation functions. Moreover, the activation function with the highest AdAct weight in a layer can be chosen as the optimal one for that layer in the optimized architecture. To determine the additional parameters from the AdAct module, consider the 7 weights in $\mathbf{w}$ and one from PReLU, yielding $8$ per unit. Then the additional parameter burden, $N_{\text{AdAct}}$, attributable to the AdAct activation units is given by $N_{\text{AdAct}} = 8n_{\text{AdAct}}$ where $n_{\text{AdAct}}$ is the number of AdAct units.

## 3 EXPERIMENTS AND RESULTS

| Model | FMNIST | CIFAR10 | SVHN | FER |
|---|---|---|---|---|
| ConvNet-AdAct | 0.921 | 0.766 | 0.911 | 0.546 |
| ConvNet-ReLU | 0.918 | 0.759 | 0.913 | 0.550 |
| ConvNet-Optimized | 0.922 | 0.766 | 0.917 | 0.565 |

Table 1: Performance comparison of ConvNet with different activation functions.

In our study, we employed several variants of a ConvNet architecture, each differentiated by its activation function, while all other architectural aspects remained constant. The ConvNet architecture and the training details are described in Appendix A.1 and A.2 respectively. For a graphical illustration of the weights, we direct the reader to refer to Appendix B.1. The ConvNet-AdAct model demonstrates enhanced performance on FMNIST and CIFAR10 compared to ConvNet-ReLU, while the ConvNet-Optimized version shows superior results on both SVHN and FER datasets, indicating its effectiveness in handling complex image recognition tasks. These outcomes underscore the importance of selecting optimal activation functions to boost neural network accuracy across various types of visual data [1].

## 4 CONCLUSION

The AdAct module integrates diverse activation functions into a single learnable framework, and aids in choosing optimal activation functions, underscoring its potential in neural architecture optimization. Future work will explore AdAct's utility in wider computer vision and natural language processing tasks, including object detection, image segmentation, and transformer models.

[1] Appendix B.2 further provides a detailed discussion of the performance results.

### URM STATEMENT

We acknowledge that all authors of this work meets the URM criteria of ICLR 2024 Tiny Papers Track.

## REFERENCES

Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Delving deep into rectifiers: Surpassing human-level performance on imagenet classification. In *Proceedings of the IEEE international conference on computer vision*, pp. 1026–1034, 2015.

Ilya Loshchilov and Frank Hutter. Decoupled weight decay regularization. In *International Conference on Learning Representations*, 2019. URL https://openreview.net/forum?id=Bkg6RiCqY7.

Ningning Ma, Xiangyu Zhang, Ming Liu, and Jian Sun. Activate or not: Learning customized activation. In *2021 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 8028–8038, 2021a. doi: 10.1109/CVPR46437.2021.00794.

Ningning Ma, Xiangyu Zhang, Ming Liu, and Jian Sun. Activate or not: Learning customized activation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 8032–8042, June 2021b.

Diganta Misra. Mish: A self regularized non-monotonic activation function, 2020.

Chigozie Nwankpa, Winifred Ijomah, Anthony Gachagan, and Stephen Marshall. Activation functions: Comparison of trends in practice and research for deep learning, 2018.

Prajit Ramachandran, Barret Zoph, and Quoc V. Le. Searching for activation functions, 2017.

Bing Xu, Naiyan Wang, Tianqi Chen, and Mu Li. Empirical evaluation of rectified activations in convolutional network, 2015.

Barret Zoph and Quoc V. Le. Neural architecture search with reinforcement learning, 2017.

## A APPENDIX A: CONVNET ARCHITECTURE AND TRAINING DETAILS

### A.1 CONVNET ARCHITECTURE

The ConvNet architecture, implemented using the PyTorch framework, consists of 4 convolutional layers, each followed by batch normalization and max pooling layers, culminating in a fully connected layer. The architecture's convolutional layers are detailed in Table 2.

| Layer | In Channels | Out Channels | Kernel Size |
|---|---|---|---|
| 1 | 1 (Grayscale); 3 (RGB) | 32 | 3 |
| 2 | 32 | 64 | 3 |
| 3 | 64 | 128 | 3 |
| 4 | 128 | 256 | 3 |

Table 2: Convolutional layers in the ConvNet architecture.

Following each convolutional layer, batch normalization is applied to stabilize and accelerate the training process. The batch normalization layers correspond to the output channels of each convolutional layer. After batch normalization, max pooling with a kernel size of 2 and a stride of 2 is used to reduce the spatial dimensions of the feature maps.

The activation functions employed by each convolutional layer vary depending on the model variant being used. The architecture concludes with a fully connected layer designed to output logits for classification. This layer's design is directly dependent on the number of classes and the dimensions of the feature maps produced by the preceding convolutional and pooling layers.

The weights of the convolutional layers are initialized via Kaiming uniform initialization (He et al., 2015). The models were trained using the Adam optimizer with weight decay (Loshchilov & Hutter, 2019), with a learning rate of $10^{-3}$ and utilizing the cross-entropy loss.

Three variants of the base ConvNet architecture are employed, which include ConvNet-AdAct, utilizing the adaptive activation (AdAct) module, ConvNet-ReLU with the conventional ReLU activation, and ConvNet-Optimized, which incorporates the most effective activation functions as identified by the AdAct analysis.

### A.2 Training Details

In our comparative analysis, we leverage both benchmark datasets and application-specific collections to evaluate the robustness of our model. CIFAR10 and FashionMNIST serve as standard benchmarks in machine learning for image classification; CIFAR10 with $60,000$ $32 \times 32$ color images across $10$ classes of real-world objects, and FashionMNIST with $70,000$ $28 \times 28$ grayscale images across $10$ fashion-related categories. Complementing these, the Street View House Numbers (SVHN) and the Facial Emotion Recognition (FER) datasets offer more real-life scenario-oriented data; SVHN presents a variety of real-world digit images from Google Street View with $73,257$ training and $26,032$ testing samples, while FER provides $48 \times 48$ pixel grayscale images of faces, where we have partitioned the original $28,709$ examples into an $80\%$ training and $20\%$ testing split. These diverse datasets facilitate a comprehensive evaluation of convolutional network performance across different domains of visual recognition tasks.

For both datasets, standard preprocessing including normalization, with a mean and standard deviation of $0.5$, was applied to ensure consistent and efficient training. A batch size of $64$ images is used for training while for testing, a batch of $1000$ images is used. The model was trained over $15$ epochs, and the model was evaluated during training and was conducted every epoch using the test set. The best model weights, determined by the highest test accuracy, were saved for future reference, enabling the retrieval of the most effective version of the model post-training. The highest test accuracy is reported for analyzing the performance metrics.

## B Experimental Results

### B.1 Graphical Illustration of Activation Function Weights

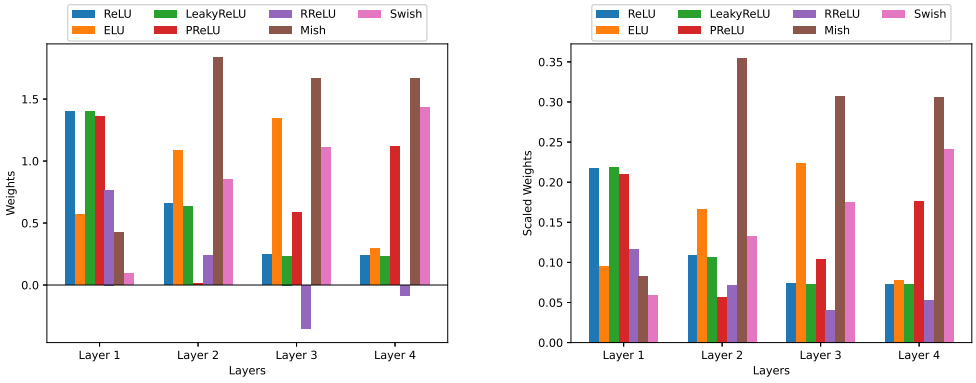


Figure 1: Weights and softmax-scaled weights of AdAct modules of ConvNet trained on FMNIST.

The figures above showcase the weights and softmax-scaled weights of the AdAct modules within the ConvNet architecture, following training over 15 epochs. Figure 1 illustrates the results for the network trained on the FMNIST dataset, while Figure 2 displays the outcomes for the CIFAR10 dataset.

For FMNIST, as depicted in Figure 1, the LeakyReLU activation function exhibits a dominant weight in the first layer, suggesting its significant role in initial feature activation. Conversely, the RReLU

function shows notably lesser importance across all layers, with negative weights in the later layers, indicating a potential detrimental effect or lack of contribution to the model's learning process. As the network depth increases, the Mish activation function becomes increasingly prominent, particularly in the final layers, which implies its effectiveness in improving the Convnet performance.

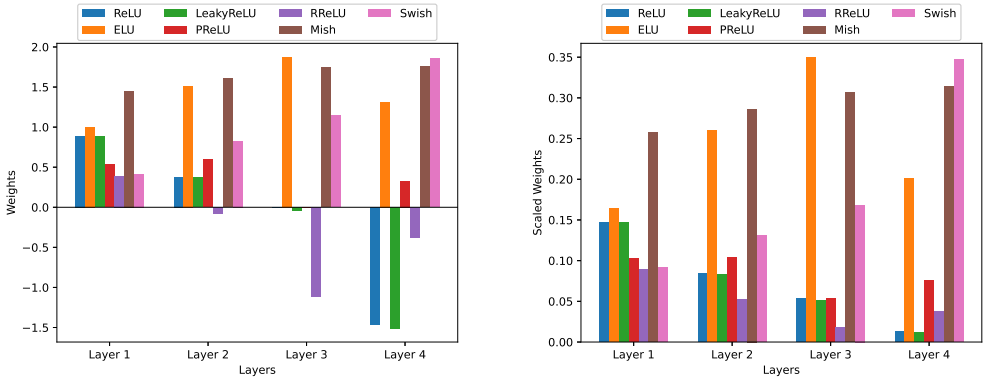


Figure 2: Weights and softmax-scaled weights of AdAct modules of ConvNet trained on CIFAR10.

In the case of CIFAR10, shown in Figure 2, Mish activation maintains a consistent lead in the initial layers, which may be attributed to its ability to handle the more complex color image data within CIFAR10. The third layer's preference shifts towards the ELU activation function, while the fourth layer displays a strong inclination for Swish. This suggests that these functions may offer advantages in processing the higher-level features that are characteristic of deeper network layers. Notably, activations like ReLU, LeakyReLU, RReLU, and PReLU demonstrate small or negative weights in certain layers, particularly in the third or fourth layer, which might reflect its incompatibility with the specific feature representations at that depth resulting in a less prominent contribution to the learning process of the Convnet.

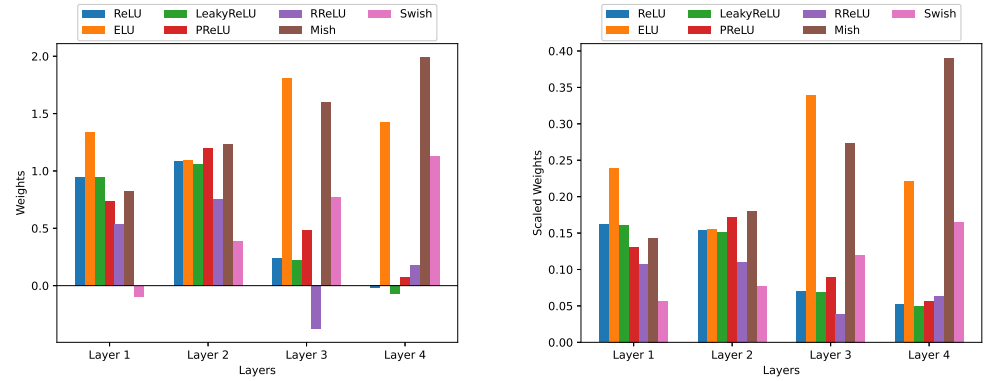


Figure 3: Weights and softmax-scaled weights of AdAct modules of ConvNet trained on SVHN.

For SVHN, as portrayed in Figure 3, the Mish and ELU activation functions exhibit considerable weights in all layers, with ELU and MIST having a pronounced dominance in the third and fourth layers, underscoring their substantial role in intricate feature extraction for numeral classification from real-world images. Contrastingly, the other activations like ReLU and LeakyReLU manifest small or negative weights in specific layers, notably in the latter stages, potentially reflecting their lesser suitability for learning the feature representations as required by the SVHN dataset.

In the case of FER, illustrated in Figure 4, Mish dominates in the first layer, suggesting its effectiveness in capturing the initial emotional features from facial expressions. The second and third layers show a strong inclination towards ELU, indicating its potential in processing and refining the emotional features extracted. Finally, in the last layer, Mish is the highest-weighed activation, potentially

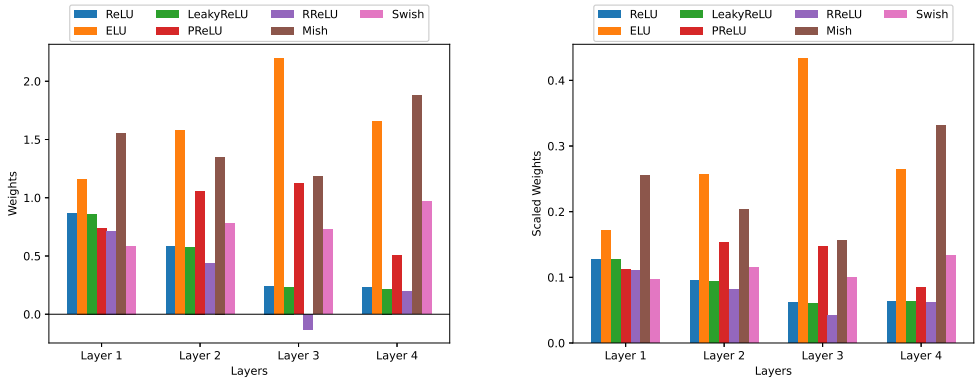


Figure 4: Weights and softmax-scaled weights of AdAct modules of ConvNet trained on FER.

due to its ability to assist in the higher-level abstraction and recognition of complex emotional states from facial cues.

### B.2 Performance Metrics of ConvNet Variants

Performance-wise, the ConvNet-AdAct models demonstrated a nuanced superiority as illustrated in Table 1. For FashionMNIST (FMNIST), the ConvNet-AdAct achieved a slightly better test accuracy than that of ConvNet-ReLU. The ConvNet-Optimized configuration for FMNIST further improved the accuracy. In the CIFAR10 dataset, the ConvNet-AdAct model again outperforms the ConvNet-ReLU. For the Street View House Numbers (SVHN) dataset, the ConvNet-Optimized model outperforms the ConvNet-ReLU model, indicating its effectiveness in complex tasks like digit classification, and a similar result is observed in the Facial Expression Recognition (FER) dataset, showcasing its capability in nuanced challenges like facial expression recognition.

These results provide a comprehensive overview of how varying activation functions impact the performance of a standard convolutional neural network architecture, emphasizing the effectiveness of the AdAct module in optimizing network performance. Furthermore, it can also be noted that ConvNet-optimized no longer needs to incur the overhead of the additional weights of the AdAct module while performing on par or slightly better than ConvNet-AdAct as it utilizes the most effective activations identified with AdAct analysis.