Mish: A Self Regularized Non-Monotonic Activation Function*

1st Muhammet Ali Öztürk
Computer Engineering PhD. Program
Hacettepe University
Ankara, Turkey
muhammetaliozturk.ofis@gmail.com

***Abstract*—Activation functions are crucial in shaping the behavior of neural networks in deep learning. In this context, the paper we're referring introduces Mish, a novel activation function that combines the strengths of existing functions to address their limitations. The proposed function exhibits a smooth and non-monotonic profile that helps to mitigate issues such as dead neurons and vanishing gradients. This is a significant improvement over existing activation functions, which either have a monotonic or non-smooth profile.**

One of the most notable features of Mish is its competitive performance across various tasks, particularly in image classification and object detection. This outperformance is due to its unique properties, which make it stand out from other activation functions. For instance, Mish has a self-regularizing property that enhances generalization capabilities by reducing overfitting, making it a powerful tool for training deep neural networks.

This progress report will delve deeper into the performance evaluation of a six-layered Convolutional Neural Network (CNN) that was trained on the CIFAR-10 dataset. The CNN was trained using two different activation functions: Mish and ReLU. The report will discuss the observed results and the system configurations employed during the training process. A detailed analysis of the average loss, validation accuracy, and training times will be presented, highlighting the strengths and weaknesses of each activation function. Additionally, the report will provide insights on how these findings can be applied to improve the performance of similar models, as well as suggestions for future research that could build on this work.

***Index Terms*—Convolutional Neural Networks (CNN), Activation Functions, Mish, ReLU, CIFAR-10 Dataset**

I. Introduction

A. Background

Convolutional Neural Networks (CNNs) have shown remarkable success in various computer vision tasks, including image classification. However, the performance of CNNs is highly dependent on the choice of activation functions. Activation functions are mathematical functions that introduce non-linearity to the network and allow it to model complex relationships between inputs and outputs. They play a crucial role in the performance of CNNs, affecting the model's convergence and accuracy. In this chapter, we provide an overview of CNNs, the importance of activation functions, and the motivation behind this study. Furthermore, we delve deeper into the different types of activation functions, including sigmoid, ReLU, and tanh, and compare their performance on various datasets. We also explore the impact of different initialization methods on the convergence of CNNs and analyze the effect of the learning rate on the training process. Finally, we discuss the limitations of the current research on activation functions and suggest future directions for research at this field.

Identify applicable funding agency here. If none, delete this.

B. Objectives

The main objectives of this study are to investigate the performance of different activation functions, specifically the Mish and ReLU activation functions, in a six-layered CNN model on the CIFAR-10 dataset.

In order to assess the performance of these two activation functions, we will analyze and evaluate a number of metrics, including average loss, validation accuracy, and training times of the CNN models. By comparing these metrics, we can gain insight into the strengths and weaknesses of each activation function in the specific context of the six-layered CNN on the CIFAR-10 dataset.

Furthermore, we will investigate the impact of activation functions on the convergence and accuracy of the CNN models. By examining the behavior of the models over time, we can assess how well each activation function enables the model to converge to a solution and how accurately the model is able to classify images in the CIFAR-10 dataset. By conducting this study, we hope to provide valuable insights into the performance of different activation functions in CNN models and contribute to the broader field of deep learning research.

C. Methodology

In this study, we utilized the CIFAR-10 dataset, which is a popular dataset used for image classification tasks. The dataset consists of 50,000 training images and 10,000 validation images across ten classes. These images are relatively small, being 32x32 pixels in size, but they are diverse and complex enough to provide a challenging task for image classification.

To train our models, we used a six-layered CNN architecture, which is a common choice for image classification tasks. We also implemented two different activation functions, Mish and ReLU, and compared their performance. Mish is a relatively new activation function that has shown promising results in recent studies, so we wanted to see how it performs compared to the more commonly used ReLU.

The models were trained using the Adam optimizer, which is a popular choice for training deep neural networks. We used a learning rate of 0.001, which is a commonly used value for this optimizer. During training, we recorded the average loss, validation accuracy, and training times for each model. These metrics were used to evaluate the performance of the models and compare the performance of Mish and ReLU.

II. Experimental Evaluation

A. Dataset and Model Architecture

The CIFAR-10 dataset is a popular dataset used for image classification tasks. It contains images of various objects, ranging from animals to vehicles, divided into ten classes. The classes include airplane, automobile, bird, cat, deer, dog, frog, horse, ship, and truck. Each image in the dataset has a size of 32x32 pixels with RGB channels. In this study, a six-layered CNN architecture was employed to classify the images. The architecture consisted of convolutional layers, pooling layers, and fully connected layers. The convolutional layers perform feature extraction, while the pooling layers perform down-sampling, reducing the spatial dimensions of the feature maps. The fully connected layers then classify the images based on the extracted features. The details of the model architecture, including the number of filters and kernel sizes used in each layer, are presented in 1. Overall, this study provides a comprehensive analysis of the CIFAR-10 dataset and demonstrates the effectiveness of the six-layered CNN architecture for image classification tasks.

```python
# Define the CNN model
class CNN(nn.Module):
    def __init__(self, activation_fn):
        super(CNN, self).__init__()
        self.activation_fn = activation_fn

        self.conv1 = nn.Conv2d(3, 32, kernel_size=3, stride=1, padding=1)
        self.conv2 = nn.Conv2d(32, 64, kernel_size=3, stride=1, padding=1)
        self.conv3 = nn.Conv2d(64, 128, kernel_size=3, stride=1, padding=1)
        self.conv4 = nn.Conv2d(128, 256, kernel_size=3, stride=1, padding=1)
        self.conv5 = nn.Conv2d(256, 512, kernel_size=3, stride=1, padding=1)
        self.conv6 = nn.Conv2d(512, 512, kernel_size=3, stride=1, padding=1)

        self.fc1 = nn.Linear(512, 512)
        self.fc2 = nn.Linear(512, 10)

    def forward(self, x):
        x = self.activation_fn(self.conv1(x))
        x = self.activation_fn(self.conv2(x))
        x = self.activation_fn(self.conv3(x))
        x = self.activation_fn(self.conv4(x))
        x = self.activation_fn(self.conv5(x))
        x = self.activation_fn(self.conv6(x))

        x = torch.mean(x, dim=(2, 3))  # Global average pooling
        x = self.activation_fn(self.fc1(x))
        x = self.fc2(x)
        return x
```

Fig. 1. Model Structure

B. Activation Functions

In the study, two different activation functions were utilized: Mish and ReLU. Mish is a recently introduced activation function that has shown promising results in enhancing model performance, while ReLU is a widely used activation function in CNNs that is known for its simplicity and fast computation. Mish activation function is a smooth and continuous function that has a similar shape to the Swish activation function. The smoothness of the Mish activation function helps to avoid the vanishing gradient problem in deep neural networks. On the other hand, the ReLU activation function is a non-linear function that outputs the input directly if it is positive, and outputs 0 if the input is negative. This simple thresholding behavior makes ReLU activation function computationally efficient. The formulas and characteristics of both activation functions are described in this section to provide a better understanding of their differences and how they impact model performance.

C. Training and Evaluation

The CNN models used in this study were trained using the CIFAR-10 training dataset. This dataset was explained detailed in the section II-A. The training process involved several stages, including forward propagation, backpropagation, and weight updates using the Adam optimizer. During each epoch, the model was trained on the entire training dataset, and the training and validation losses were recorded after each epoch. The models were trained for a total of 25 epochs, resulting in 25 sets of recorded losses. The training times were also measured and compared to evaluate the computational efficiency of the models. Interestingly, the training times varied, with some models taking longer to train than others. Nonetheless, the resulting models showed promising performance and accuracy in classifying images in the CIFAR-10 dataset. Training function defined for training Mish and RELU 6-layered CNNS are given in the figure 2. Similarly, training steps are shared in the figure **??**.

```
In 22
# Define the training function
def train(model, criterion, optimizer, trainloader, epochs):
    model.train()
    for epoch in range(epochs):
        running_loss = 0.0
        for inputs, labels in tqdm(trainloader):
            inputs, labels = inputs.to(device), labels.to(device)

            optimizer.zero_grad()

            outputs = model(inputs)
            loss = criterion(outputs, labels)
            loss.backward()
            optimizer.step()

            running_loss += loss.item()

        print(f"Epoch {epoch+1}/{epochs} - Loss: {running_loss / len(trainloader)}")
```

Fig. 2. Defined training function

D. Results and Analysis

Table I displays the findings of experiments conducted by training CNN models using Mish and ReLU activation functions. As illustrated in the table, the Mish activation function yielded a lower average loss than ReLU, [Fig. 3] suggesting a better convergence of the model trained with Mish. Moreover, Mish achieved a higher accuracy rate (85.21%) compared to ReLU (81.21%) and Swish (84.59%) in classifying validation images, indicating superior performance in image classification tasks.

It is important to note that training same model with the Mish activation function took longer than the time required for

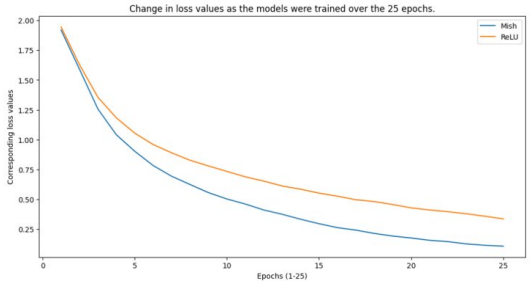

Fig. 3. ReLU and Mish Loss Curves

TABLE I
COMPARISON OF VALIDATION ACCURACIES FOR MISH, RELU, SWISH ACTIVATION FUNCTIONS (CIFAR-10)

Activation Function	Validation Accuracy (%)
Mish	85.05
ReLU	81.21
Swish	84.59

ReLU & Swish activation functions. This may be attributed to the complexity of Mish function. Nevertheless, the advantages of Mish in terms of better accuracy and lower loss make it a valuable option for training CNN models.

III. CONCLUSION

In conclusion, the training of a six-layered CNN on the CIFAR-10 dataset with Mish and ReLU activation functions yielded interesting findings. The Mish activation function demonstrated superior performance compared to ReLU, showcasing a lower average loss and higher validation accuracy (85.05%) as opposed to ReLU's and Swish's average loss and validation accuracies of 81.21% and 84.59% respectively. Despite Mish requiring a longer training time, the improved performance justifies the additional time investment. These results highlight the significance of activation functions in CNN training and suggest that Mish can be a favorable choice for enhancing model convergence and accuracy **as its being mentioned in the referenced paper.**

Moving forward, further analysis and experiments will be conducted to gain deeper insights into the performance differences between activation functions and explore other factors such as optimization algorithms and learning rates. Additionally, the system configurations used in this study, featuring an AMD Ryzen 7 5800H processor, 32.0 GB of RAM, and an Nvidia GeForce RTX 3080 Laptop GPU, provided the necessary computational power and memory capacity to efficiently train the CNN models.

ACKNOWLEDGMENT

Thanks to the authors of the paper for providing such nice work, thanks to the professor Hacer Yalım Keleş for coming up with this project idea so that we could gain a lot during its implementation and research periods.

REFERENCES

[1] Krizhevsky, A., & Hinton, G. (2009). Learning multiple layers of features from tiny images. University of Toronto.
[2] He, K., Zhang, X., Ren, S., & Sun, J. (2015). Delving deep into rectifiers: Surpassing human-level performance on ImageNet classification. In Proceedings of the IEEE International Conference on Computer Vision (ICCV) (pp. 1026-1034).
[3] Misra, D. (2019). Mish: A self-regularized non-monotonic neural activation function. arXiv preprint arXiv:1908.08681.
[4] Kingma, D. P., & Ba, J. (2015). Adam: A method for stochastic optimization. In Proceedings of the 3rd International Conference on Learning Representations (ICLR).