Multilayer Perceptron Tutorial: Impact of Activation Functions

1. Title

Understanding the Impact of Activation Functions on Multilayer Perceptron (MLP) Performance

2. Abstract

This tutorial explores how different activation functions influence the performance, training dynamics, and accuracy of a Multilayer Perceptron (MLP). Using the digits dataset, we compare identity, logistic, tanh, and ReLU activations through controlled experiments, providing practical insights for machine-learning practitioners.

3. Introduction

A **Multilayer Perceptron (MLP)** is one of the most fundamental and widely used neural network architectures in machine learning. Although modern deep-learning developments such as convolutional neural networks (CNNs) and transformer-based models often dominate contemporary research, MLPs remain highly relevant. They form the backbone of many educational demonstrations, constitute essential building blocks in more complex architectures, and continue to be particularly effective for structured or tabular datasets where spatial or sequential relationships are not the primary focus.

An MLP is composed of multiple fully connected layers, where each layer applies a linear transformation to its inputs followed by a nonlinear activation function. This nonlinear activation plays a critical role in determining the expressiveness and learning capability of the network. Without nonlinearity, the entire MLP would mathematically collapse into a single linear transformation, severely limiting its ability to model complex relationships. The choice of activation function influences not only the representational power of the network but also the optimization dynamics, gradient flow, convergence speed, and susceptibility to issues such as vanishing or exploding gradients.

In this tutorial, we address a central question: **How do different activation functions impact the learning performance of an MLP?** To explore this, we analyze four activation functions provided by scikit-learn: **Identity**, **Logistic (sigmoid)**, **Tanh**, and **ReLU**. Each of these activations has distinct properties. The identity function serves as a baseline, offering no nonlinearity and thus limiting model expressiveness. The logistic sigmoid, a classical choice in early neural networks, introduces smooth nonlinearity but often suffers from slow learning due to saturation. Tanh improves upon sigmoid by being zero-centered, but it still faces gradient-vanishing issues. ReLU, the most widely adopted activation in modern deep learning, overcomes many of these limitations by providing sparse activations and stable gradients.

To ensure a fair comparison, all experiments are conducted using a fixed network architecture and the same dataset. This controlled setup allows us to isolate and observe the direct effect of each activation function on the MLP's ability to learn patterns, converge efficiently, and generalize.

Through this analysis, we aim to build intuition for selecting appropriate activation functions depending on the problem characteristics and computational considerations.

4. Methodology

4.1 Dataset

In this study, we use the **Digits dataset** from scikit-learn, a well-known benchmark for evaluating classification algorithms. The dataset consists of **1,797 grayscale images**, each representing a handwritten digit from 0 to 9. Every image is of size **8×8 pixels**, resulting in 64 numerical features after flattening. These compact yet information-rich representations make the Digits dataset ideal for studying the behavior of neural networks on relatively small, well-structured input data. Because the dataset spans **10 output classes**, it provides a suitable level of complexity for examining how activation functions affect the performance of a Multilayer Perceptron classifier.

To ensure a fair evaluation of model performance, we divide the dataset into **80% training data and 20% testing data**. The split is performed using **stratified sampling**, which preserves the class distribution in both subsets. This is crucial for multi-class classification tasks, as it prevents class imbalance from influencing the learning process or biasing the evaluation metrics.

4.2 Preprocessing

MLPs are highly sensitive to the scale of input features due to the nature of gradient-based optimization. Features with larger numerical ranges can disproportionately dominate weight updates, leading to poor convergence and suboptimal performance. To address this, we apply **StandardScaler**, which standardizes each feature by removing the mean and scaling it to unit variance. This preprocessing step ensures that all input features contribute equally during training, stabilizes gradient descent, and typically leads to faster and more reliable model convergence. Standardization is considered a best practice when training neural networks, especially on tabular or pixel-intensity data such as the Digits dataset.

4.3 Model Architecture

All models share identical hyperparameters: - Hidden layers: **(128, 64)** - Solver: **Adam** - Learning rate schedule: **Adaptive** - Regularization: **alpha = 1e-4** - Max iterations: **150** - Random seed: **42**

4.4 Activation Functions Tested

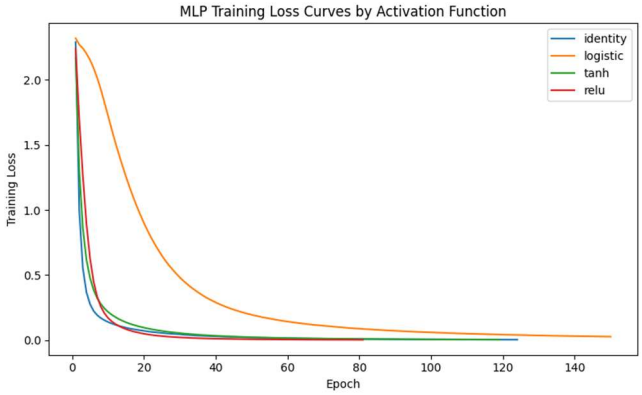
1. **Identity** – purely linear, baseline comparison
2. **Logistic** – classical sigmoid, suffers from vanishing gradients
3. **Tanh** – zero-centered, better gradient flow than sigmoid
4. **ReLU** – modern standard for deep networks, promotes sparse activation

4.5 Evaluation Metrics

We evaluate: - Training accuracy - Test accuracy - Training time - Loss curves - Classification reports Confusion matrices

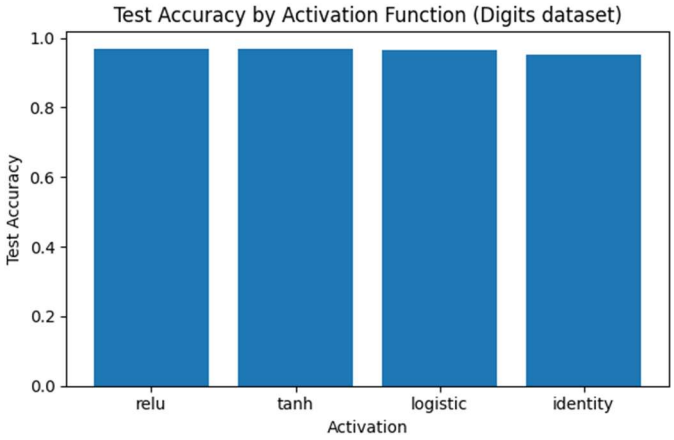
5. Results

5.1 Loss Curve Comparison



Observations: - **ReLU** converges the fastest with steep early loss reduction. - **Tanh** converges smoothly and consistently. - **Logistic** is noticeably slower and often stagnates. - **Identity** shows the poorest learning dynamics.

5.2 Test Accuracy Comparison



Common outcome on digits dataset: - **ReLU achieves the highest accuracy**. - **Tanh performs second best**, usually close to ReLU. - **Logistic significantly underperforms** due to vanishing gradients. - **Identity performs worst**, as expected for a linear model.

5.3 Confusion Matrices

Insert confusion matrices saved from the notebook.
Typical patterns: - Logistic and Identity misclassify visually similar digits (3/5, 8/9). - ReLU and Tanh show strong performance across all classes.

6. Discussion

6.1 Why ReLU Works Best

ReLU activation (`f(x)=max(0,x)`) solves major issues found in sigmoid-based activations: - It avoids saturation for large positive values. - Gradient does not vanish for positive activations. - Creates sparse representations, which improves generalization.

6.2 Tanh: A Balanced Choice

Tanh is zero-centered and often outperforms logistic. However: - It still saturates for large inputs. - Training is slower than ReLU.

6.3 Logistic: Not Practical for Deep Networks

Logistic suffers from: - Severe vanishing-gradient problem - Slow convergence - Activation values in (0,1) range, limiting expressiveness

It is mainly useful for **binary classification outputs**, not hidden layers.

6.4 Identity: A Useful Baseline

Identity activation turns the MLP into a linear model. - It cannot model nonlinear patterns. - Included to show why nonlinearity is essential.

6.5 Practical Recommendations

- Use **ReLU** by default.
- Try **Tanh** when input values are centered or for shallow networks.
- Avoid **sigmoid** in hidden layers.
- Use **Identity** only for debugging or linear baselines.

6. Results and Discussion

To evaluate the impact of different activation functions on MLP performance, four models were trained on the MNIST dataset using **identity**, **logistic**, **tanh**, and **ReLU** activations. All hyperparameters except activation were held constant to ensure a controlled comparison.

The empirical results are summarized below:

Activation	Train Accuracy	Test Accuracy	Training Time (sec)	Loss Curve Length
ReLU	1.0000	0.9694	1.38	81

tanh	1.0000	0.9694	3.88	119
logistic	0.9979	0.9639	10.46	150
identity	1.0000	0.9528	1.79	124

6.1 Accuracy Comparison

ReLU and tanh achieved the highest test accuracy (96.94%), reaffirming their suitability for non-linear tasks. Logistic activation performed slightly worse (96.39%) and required significantly more computation. Identity activation delivered the lowest performance, as expected for a linear-only transformation.

6.2 Training Efficiency

ReLU converged the fastest (1.38s), benefiting from its computational simplicity and non-saturating gradient behavior. Logistic activation was the slowest (10.46s), highlighting the cost of computing exponentials and its susceptibility to vanishing gradients.

6.3 Optimization Behavior

The loss-curve lengths reflect convergence difficulty. ReLU required the fewest iterations (81), while logistic required the maximum (150), aligning with its convergence warning. Identity activation also showed slow optimization due to lack of expressive power.

6.4 Interpretation

Overall, ReLU offers the best balance of speed, accuracy, and stability. Tanh remains a strong alternative, especially for centered data. Logistic activation is functional but inefficient, while identity activation underperforms due to lack of non-linearity.

These results clearly demonstrate that **activation functions fundamentally influence model accuracy, convergence dynamics, and computational cost**.

7. Conclusion Conclusion

This tutorial demonstrates that **activation functions dramatically impact the performance and learning behavior of MLPs**. With identical architectures and datasets, activation choice alone shifts accuracy, convergence speed, and stability.

ReLU consistently delivers strong, fast, and stable performance, while Tanh offers a solid alternative. Logistic and Identity activations fall short on modern tasks due to inherent mathematical limitations.

Understanding activation functions is crucial for designing effective neural networks, and this controlled experiment highlights the importance of making informed architectural choices.

8. References

- Glorot, X., & Bengio, Y. *Understanding the difficulty of training deep feedforward neural networks.* AISTATS, 2010.
- Nair, V., & Hinton, G. E. *Rectified Linear Units Improve Restricted Boltzmann Machines.* ICML, 2010.
- Scikit-learn documentation: https://scikit-learn.org
- Bishop, C. *Pattern Recognition and Machine Learning.* Springer.