Multilayer Perceptron’s: The Impact of Depth and Width on Performance

Introduction

Multilayer Perceptron’s (MLPs) are a foundational type of neural network, commonly used for regression and classification tasks. The design of an MLP involves critical architectural choices, particularly regarding its depth (number of hidden layers) and width (number of neurons per hidden layer). These parameters directly influence the network's capacity to learn from data, generalize to unseen data, and balance computational efficiency.

The flexibility of MLPs allows them to be applied across diverse domains, such as predictive modelling in real estate, image recognition, and financial forecasting. However, improper design can lead to either underutilization of data or overfitting, making it crucial to understand the dynamics of depth and width. By analysing their impact, we aim to develop a balanced model that performs optimally without unnecessary complexity.

This tutorial explores how varying depth, and width affects an MLP's performance. We will use the “California Housing Prices” dataset to demonstrate these effects, presenting both theoretical insights and experimental results. By the end, you will understand how to design an MLP that achieves optimal performance for a given task.

Personal Reflection

Through my journey in learning about neural networks, understanding the trade-offs between depth and width became pivotal. It was particularly eye-opening to see how deeper architectures could model hierarchical patterns, while wider layers excelled at capturing diverse feature interactions. This tutorial reflects my attempts to balance these factors and aims to simplify the concepts for learners.

Background Theory

What is an MLP?

An MLP consists of:

- Input Layer: Receives raw features and passes them to the network.
- Hidden Layers: Composed of neurons applying activation functions (e.g., ReLU, Sigmoid) to learn hierarchical representations. The number of hidden layers determines the network's depth.
- Output Layer: Outputs predictions, either as continuous values for regression tasks or probabilities for classification tasks.

MLPs use a supervised learning approach, where the network adjusts its weights using a loss function to minimize prediction errors. Backpropagation and optimizers like Adam or SGD are employed to update weights iteratively. Backpropagation works by calculating the gradient of the loss function with respect to each weight using the chain rule, allowing for precise adjustments in multilayer architectures.

Depth vs Width:

- **Depth**: Refers to the number of hidden layers. More layers allow the network to model complex, hierarchical patterns in data. For example, in image recognition, deeper networks capture low-level features like edges in early layers and high-level features like objects in later layers.
- **Width**: Refers to the number of neurons in each hidden layer. Wider layers increase the network's capacity to capture diverse features within each level, enabling better representation of interactions among features.

Depth and Width Trade-offs

- Advantages of Greater Depth:
- Enhances hierarchical feature learning.
- Useful for tasks with inherent layered structures, like language models or image analysis.
- Advantages of Greater Width:
- Increases the feature representation capacity of individual layers.
- Suitable for datasets where features interact extensively at a single level.

Backpropagation and Weight Optimization

The power of MLPs lies in their ability to adjust weights efficiently through backpropagation. Key steps include:

- Forward Propagation: Data passes through the layers, producing predictions.
- Loss Calculation: The difference between predictions and actual values is quantified using a loss function (e.g., Mean Squared Error).
- Backward Propagation:
    - Gradients of the loss function with respect to weights are calculated using the chain rule.
    - Gradients are propagated backward, layer by layer.
- Weight Update: Optimizers like Adam use gradients to update weights and minimize the loss.

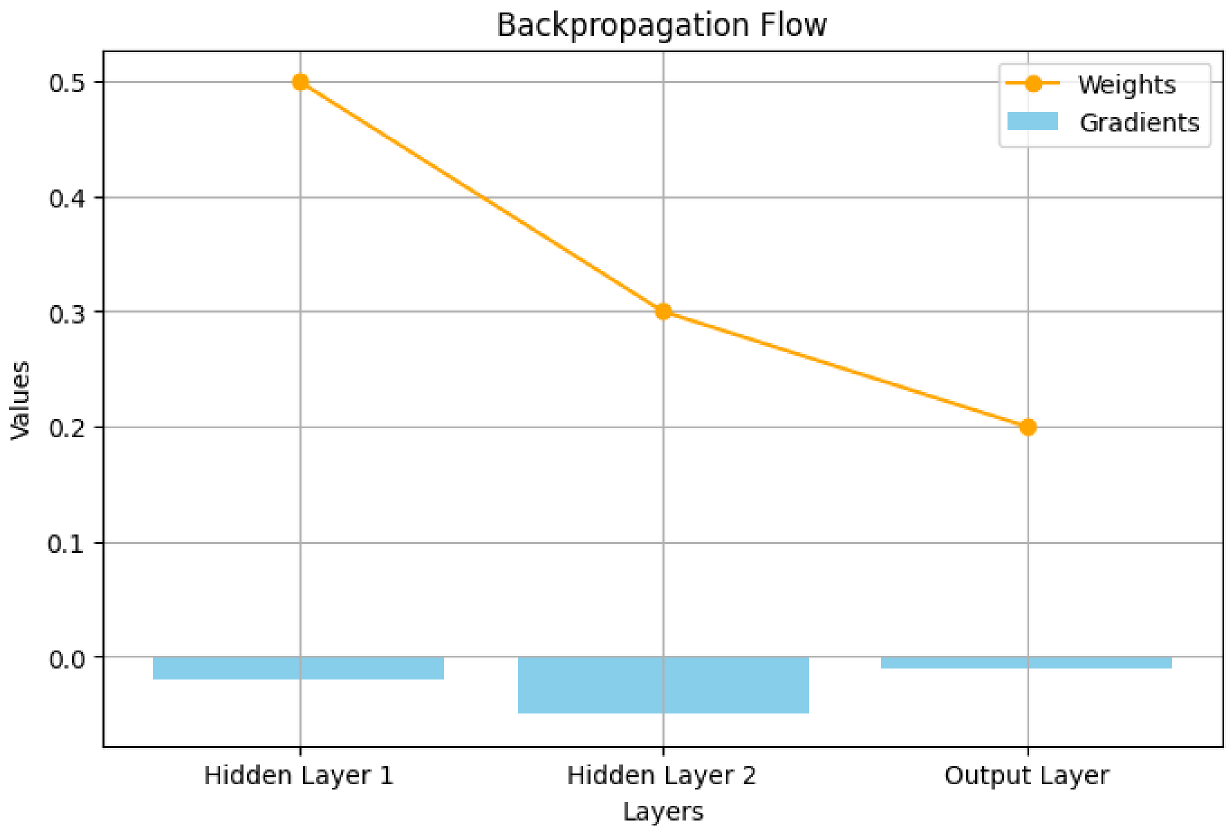

Figure1: Visualization of Backpropagation Flow

This figure shows how gradients and weights change across layers during backpropagation. Larger gradients in the initial layers indicate significant adjustments, while smaller gradients in deeper layers suggest fine-tuning. This highlights the nuanced process of weight optimization during training.

Loss Landscape and Optimization

Understanding how optimizers navigate the loss landscape provides insight into weight adjustment:

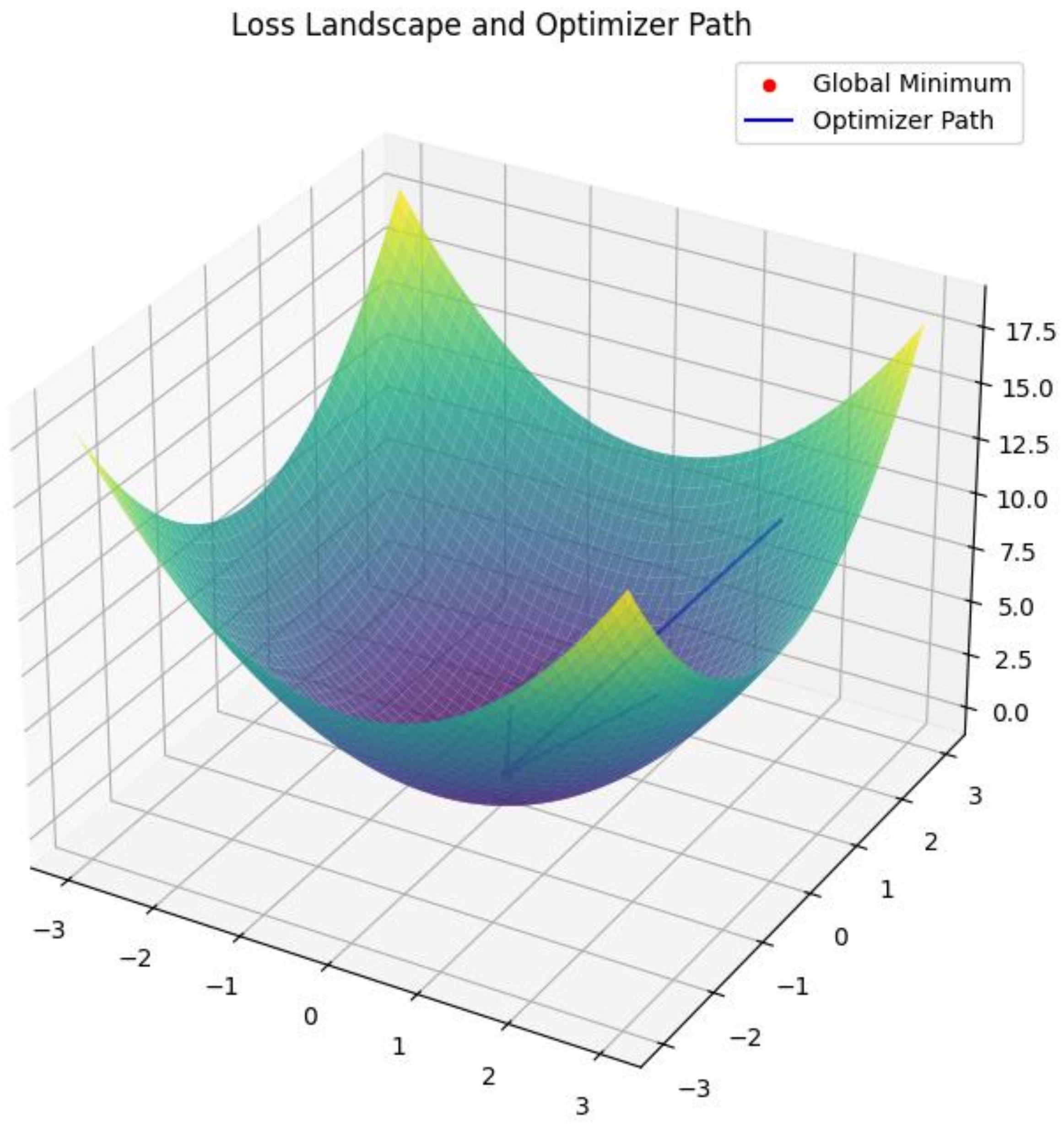

Figure 2: Loss Landscape Navigation

A 3D surface plot illustrating a loss landscape, with an optimizer path moving toward the global minimum. This visualization demonstrates how optimizers like Adam adjust weights to minimize loss. The path toward the global minimum reflects iterative updates, showcasing how gradient descent navigates complex loss surfaces.

Why Backpropagation and Optimization Matter

These concepts are essential for understanding how neural networks learn. Without backpropagation, networks cannot effectively update their weights, and without proper optimization, they may converge to suboptimal solutions or fail to converge entirely. By grasping these mechanics, practitioners can design networks that train efficiently and generalize well.

Bias-Variance Trade-Off

The bias-variance trade-off plays a critical role in MLP performance:

- High Bias: Models with insufficient depth or width may underfit the data, failing to capture its complexity and leaving significant error in predictions.
- High Variance: Overly deep or wide models may memorize training data, leading to poor generalization on unseen data.

Regularization Techniques

To manage overfitting in deep or wide networks:

- Dropout: Randomly deactivates neurons during training, reducing dependency on specific neurons.
- Early Stopping: Halts training when validation performance no longer improves.
- L2 Regularization: Adds a penalty to the loss function proportional to the magnitude of weights.

Dataset and Experiment Setup

Dataset: California Housing Prices

The dataset includes features such as median income, housing age, and average rooms per household, with the target variable being the median house value. This regression problem is ideal for exploring the impact of depth and width due to its moderate complexity.

Dataset Context

- The “California Housing Prices” dataset contains 8 numerical features and a continuous target variable.
- Key Features: Median income (strongest correlation with target), housing age, and average rooms.
- Relevance: The dataset’s moderate complexity makes it ideal for demonstrating MLP performance without requiring excessive preprocessing.

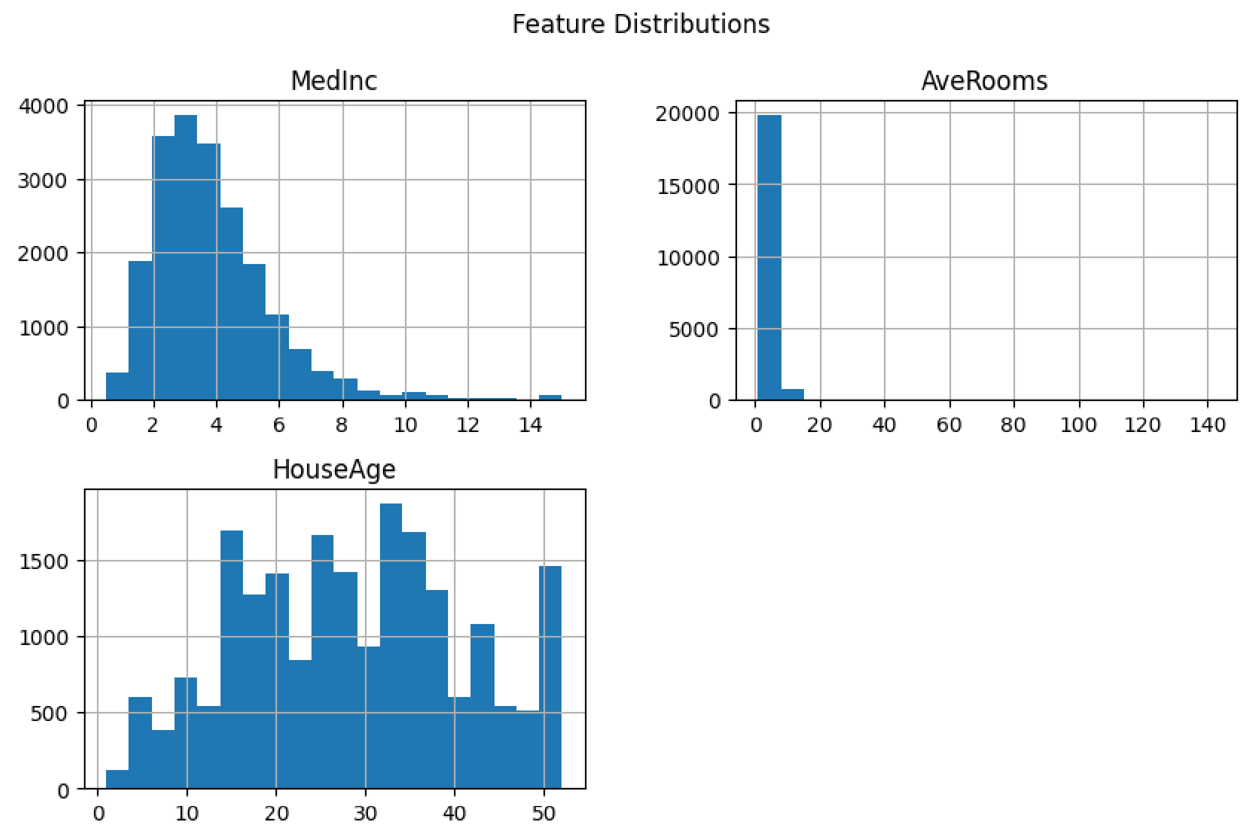

Figure 3: Feature Distribution

Experiment Design and Results

Experiment Setup - We conducted experiments with MLPs configured as follows:

- Depth: 1, 3, and 5 hidden layers.
- Width: 32, 64, and 128 neurons per layer.
- Advanced Metrics
- Early Stopping: Applied to prevent overfitting by halting training when validation loss stagnates.
- Evaluation Metrics:
- Mean Squared Error (MSE): Measures average squared difference between predicted and actual values.
- R-squared: Indicates the proportion of variance in the target explained by the model.

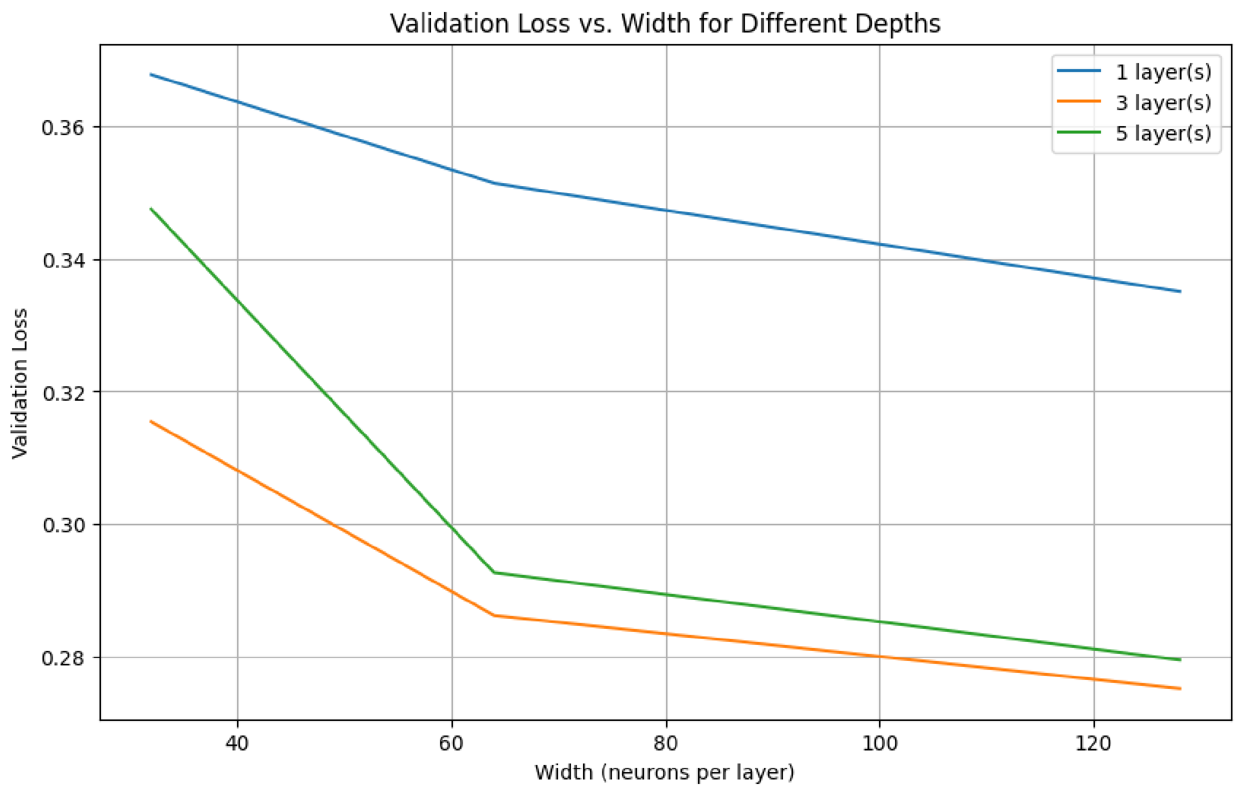

Figure 4: Validation Loss vs Width for Different Depths

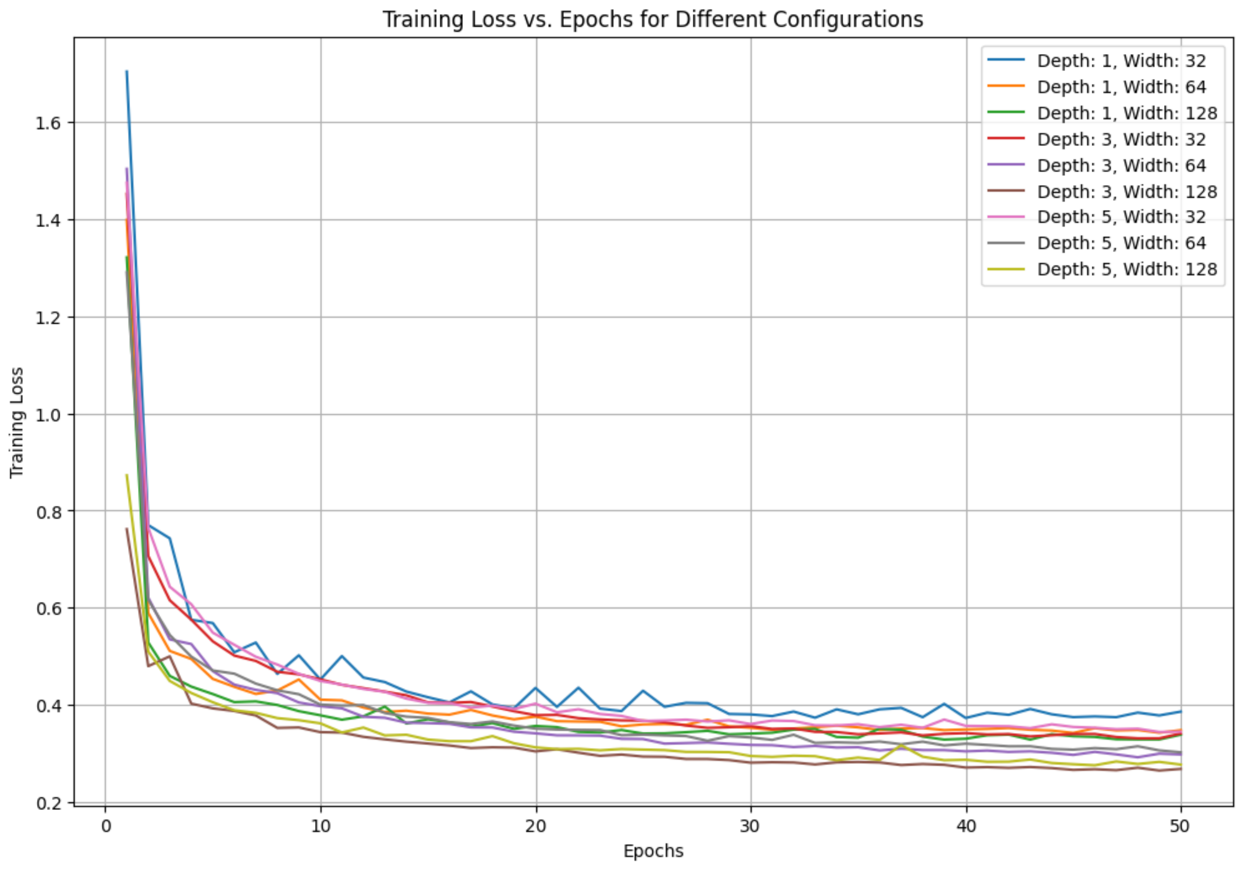

Figure 5: Training Loss vs. Epochs for Different Configurations

Observations

Effect of Depth:

- 1-layer networks: High validation loss, indicating underfitting. Validation loss decreases as width increases. Indicates that increasing width in shallow networks improves capacity but is limited in hierarchical representation.
- 3-layer networks: Optimal performance, achieving low validation loss without overfitting. Achieve the lowest validation loss overall, with optimal performance at a width of 128 neurons. This depth provides a balance between capacity and hierarchical feature learning.
- 5-layer networks: Slight overfitting, as indicated by increasing validation loss. Slightly higher validation loss compared to 3-layer models for equivalent widths, suggesting mild overfitting. The trend indicates that deeper models may not significantly outperform shallower ones for this dataset due to its complexity.

Effect of Width:

Increasing width reduces validation loss initially, but gains diminish beyond a certain point (e.g., 128 neurons). For all depths, increasing width reduces validation loss, but the returns diminish beyond 128 neurons. Wider layers enhance representation capacity but eventually plateau in effectiveness.

General Trend:

- The training loss steadily decreases over epochs, indicating effective learning.
- Validation loss closely follows the training loss, confirming that the model generalizes well without significant overfitting.
-

Optimal Configuration:

- **Validation Loss:** The final validation loss stabilizes at approximately 0.296, and the corresponding test MSE is 0.284, with an R-squared of 0.783. This suggests a well-fitted model.

Epoch Behavior:

- Early epochs (1–10) show rapid decreases in both training and validation losses.
- Later epochs see slower improvements, showing diminishing returns as the model converges.

Conclusion

This tutorial demonstrated how depth, width, backpropagation, and optimization impact MLP performance. Through experiments and theoretical exploration, we identified:

1. **Depth:** Increases hierarchical representation, allowing models to capture complex patterns, but can lead to overfitting if excessive.

2. **Width:** Enhances feature representation but exhibits diminishing returns beyond a certain capacity (e.g., 128 neurons).
3. **Regularization and Optimization:** Techniques like dropout, early stopping, and proper weight initialization are crucial for achieving a balance between bias and variance.

By understanding these principles, we can tailor MLP architectures to their specific datasets and tasks, achieving better generalization and performance while avoiding common pitfalls. This knowledge bridges the gap between theoretical neural network concepts and real-world applications, empowering individuals to build more robust and efficient models.

Finally, this exploration highlights the importance of empirical validation and iterative design in machine learning workflows. Balancing depth and width, guided by sound regularization practices, is key to developing impactful neural networks.

References

1. Goodfellow, I., Bengio, Y., & Courville, A. (2016). *Deep Learning*. MIT Press.
2. Géron, A. (2019). *Hands-On Machine Learning with Scikit-Learn, Keras & TensorFlow*. O’Reilly Media.
3. Dutta, S. (2023). *Multi-Layer Perceptron and Backpropagation: A Deep Dive*. Medium. [https://medium.com/@sanjay_dutta](https://medium.com/@sanjay_dutta/multi-layer-perceptron-and-backpropagation-a-deep-dive-8438cc8bcae6)

Student Name: Sravanth Baratam
Student Id: 23001152
Code: https://github.com/sravanth-space/ml_network_depth
dataset: https://archive.ics.uci.edu/dataset/186/wine+quality