2026 Volume: 02 Issue: 04 www.ijrpa.com ISSN 2456-9995 Research Article

International Journal Research Publication Analysis

Page: 01-10

REGULARIZATION IMPACT ON NEURAL NETWORK WEIGHT DISTRIBUTION

Dr Ramya B. N.*[1], Sanjana N. Bharadwaj[2], Sreepriya B. M.[3]

[1]Associate Professor, Department of Computer Science and Engineering, Jyothy Institute of Technology, Bengaluru, India.

[2]Department of Computer Science and Engineering, Jyothy Institute of Technology, Bengaluru, India.

[3]Department of Computer Science and Engineering, Jyothy Institute of Technology, Bengaluru, India.

Article Received: 10 March 2026
Article Revised: 30 March 2026
Published on: 20 April 2026

***Corresponding Author: Dr Ramya B. N.**
Associate Professor, Department of Computer Science and Engineering, Jyothy Institute of Technology, Bengaluru, India.

DOI: https://doi-doi.org/101555/ijrpa.3553

ABSTRACT

Regularization techniques are fundamental in improving the generalization capability of neural networks by mitigating overfitting. This paper presents a comprehensive study on the impact of different regularization strategies—L1 regularization, L2 regularization, and dropout—on neural network weight distribution and classification performance. A fully connected neural network is trained on the MNIST dataset to evaluate how each technique influences model convergence, accuracy, and internal parameter structure. Experimental results demonstrate that L1 regularization promotes sparsity in weights, L2 regularization stabilizes weight magnitudes, and dropout enhances robustness by reducing neuron co-adaptation. Weight histograms, heatmaps, and confusion matrices are analyzed to provide deeper insight into model behavior beyond traditional performance metrics. The study highlights that understanding weight distribution is crucial for designing efficient and interpretable neural networks.

KEYWORDS: Neural Networks;Regularization; L1 Regularization; L2 Regularization; Dropout;Weight Distribution; MNIST Dataset; Deep Learning; Model Generalization; Overfitting,

I. INTRODUCTION

Artificial Neural Networks (ANNs) have become a fundamental component of modern machine learning systems due to their ability to model complex and non-linear relationships within data. They are widely applied in domains such as image recognition, natural language processing, and predictive analytics. However, despite their expressive power, neural networks are highly prone to **overfitting**, especially when trained on limited or noisy datasets. Overfitting occurs when a model captures noise and irrelevant patterns from training data, resulting in poor generalization on unseen data.

To address this challenge, various **regularization techniques** have been developed to constrain model complexity and improve generalization performance. Regularization methods introduce additional constraints during training, either by modifying the loss function or altering the network architecture. Among the most commonly used approaches are **L1 regularization**, **L2 regularization**, and **dropout**.

L1 regularization encourages sparsity by penalizing the absolute values of weights, effectively reducing the number of active parameters in the model. In contrast, L2 regularization penalizes the squared magnitude of weights, leading to smaller and more stable weight values. Dropout, on the other hand, introduces stochasticity by randomly deactivating neurons during training, thereby preventing the network from relying too heavily on specific connections and improving robustness.

While many studies focus primarily on evaluating regularization techniques based on performance metrics such as accuracy and loss, limited attention has been given to understanding how these methods influence the **internal weight distribution of neural networks**. Analyzing weight distributions provides deeper insights into model behavior, interpretability, and efficiency.

In this paper, we present a detailed comparative study on the impact of L1, L2, and dropout regularization techniques on neural network performance and weight distribution. A fully connected neural network is trained on the MNIST dataset, and various evaluation tools—including loss curves, weight histograms, heatmaps, and confusion matrices—are used to analyze the effects of each method. The objective of this work is to bridge the gap between performance evaluation and internal model analysis, thereby contributing to the development of more interpretable and efficient neural network models.

II. METHODOLOGY

Dataset and Preprocessing

The MNIST dataset is used for experimental evaluation, consisting of 70,000 grayscale images of handwritten digits (0–9), each of size 28×28 pixels. The dataset is divided into 60,000 training samples and 10,000 testing samples.

To ensure efficient training and stable convergence, the following preprocessing steps are applied:

- **Normalization:** Pixel values are scaled to a range of [0,1] to improve numerical stability during gradient descent.
- **Batch Processing:** Data is loaded in batches of 64 samples to optimize computational efficiency.
- **Shuffling:** Training data is shuffled to prevent learning bias and improve generalization.

Model Architecture

A fully connected feedforward neural network is implemented to perform classification. The architecture consists of:

- **Input Layer:** 784 neurons (flattened 28×28 image)
- **Hidden Layer 1:** 128 neurons with ReLU activation
- **Hidden Layer 2:** 64 neurons with ReLU activation
- **Output Layer:** 10 neurons corresponding to digit classes

The ReLU activation function is used to introduce non-linearity and improve learning efficiency.

Regularization Techniques

To analyze the impact of regularization, three different techniques are applied:

L1 Regularization

L1 regularization adds a penalty proportional to the absolute values of weights to the loss function. This encourages sparsity by forcing less important weights toward zero.

L2 Regularization

L2 regularization penalizes the squared magnitude of weights, resulting in smaller and more evenly distributed weight values. This helps prevent extreme weight values and stabilizes

training.

Dropout Regularization

Dropout randomly deactivates neurons during training with a probability of 0.5. This prevents co-adaptation of neurons and improves the robustness of the model.

Training Configuration

The models are trained using the following configuration:

- **Optimizer:** Stochastic Gradient Descent (SGD)
- **Learning Rate:** 0.01
- **Epochs:** 5
- **Batch Size:** 64 Separate models are trained for:
- Baseline (no regularization)
- L1 regularization
- L2 regularization
- Dropout regularization

III . SYSTEM ARCHITECTURE AND DATA FLOW

The proposed system follows a structured pipeline for training and evaluating a neural network model under different regularization techniques. The architecture is designed to analyze how regularization impacts both performance and internal weight behavior.

Input Phase (Data Preparation)

- The system takes grayscale handwritten digit images of size **28×28 pixels** from the MNIST dataset.
- Each image is **flattened into a 784-dimensional vector** before being fed into the network.
- Pixel values are normalized to the range **[0,1]** to ensure stable training.

Neural Network Architecture

The model consists of a fully connected feedforward structure with multiple layers:

- **Input Layer:** Receives the flattened image (784 neurons)
- **Hidden Layer 1:** 128 neurons with ReLU activation
- **Hidden Layer 2:** 64 neurons with ReLU activation
- **Output Layer:** 10 neurons representing digit classes (0–9)

The architecture enables hierarchical feature extraction, where each layer learns progressively complex representations of input data.

Training and Learning Process

The system is trained using supervised learning with labeled input data. The training process includes:

- **Forward Propagation:** Input data passes through layers to generate predictions
- **Loss Computation:** Cross-entropy loss is calculated between predicted and actual labels
- **Backpropagation:** Gradients are computed and propagated backward
- **Weight Update:** Parameters are updated using Stochastic Gradient Descent (SGD)

Regularization Flow Integration

Regularization techniques are incorporated during the training phase as follows:

- **L1 Regularization:** Adds an absolute weight penalty to the loss function, promoting sparsity
- **L2 Regularization:** Adds a squared weight penalty, controlling weight magnitude
- **Dropout:** Randomly disables neurons during training to prevent over-dependence on specific nodes

Each technique modifies the learning dynamics, allowing comparative analysis of their effects.

Evaluation and Output

After training, the model is evaluated using test data:

- **Accuracy Measurement:** Determines classification performance
- **Confusion Matrix:** Analyzes prediction correctness across classes
- **Weight Distribution Visualization:** Includes histograms and heatmaps to study learned parameters

The system outputs both **quantitative metrics (accuracy, loss)** and **qualitative insights (weight patterns, visualization plots)**, enabling a comprehensive understanding of model behavior under different regularization techniques.

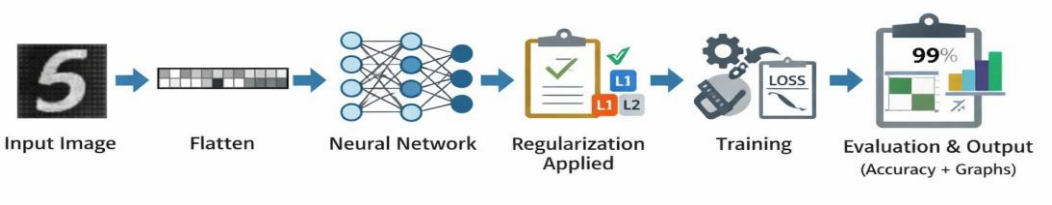


Fig 1 System Architecture Flow.

IV. RESULTS AND DISCUSSION

Performance Comparison

The classification accuracy of different models is summarized below:

Table1 Accuracy of Models

Model	Accuracy
Baseline (No Regularization)	92.93%
L2 Regularization	91.62%
L1 Regularization	**92.97%**
Dropout Regularization	92.25%

Observation:

- L1 regularization achieves the highest accuracy
- L2 slightly reduces accuracy due to stronger weight constraints
- Dropout maintains stable and competitive performance

These results indicate that moderate regularization improves generalization, while excessive constraint may slightly affect accuracy.

Loss Convergence Analysis

The training loss curves demonstrate that all models converge effectively within the given number of epochs.

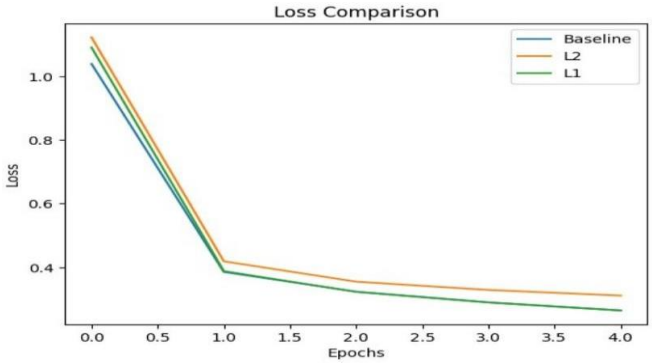


Fig.2 Loss Comparison Curve.

- Baseline and L1 models show faster convergence
- L2 exhibits slightly higher loss due to penalization
- Dropout shows smooth but slightly slower convergence

Weight Distribution Analysis

Weight distribution is analyzed using histograms generated from trained models.

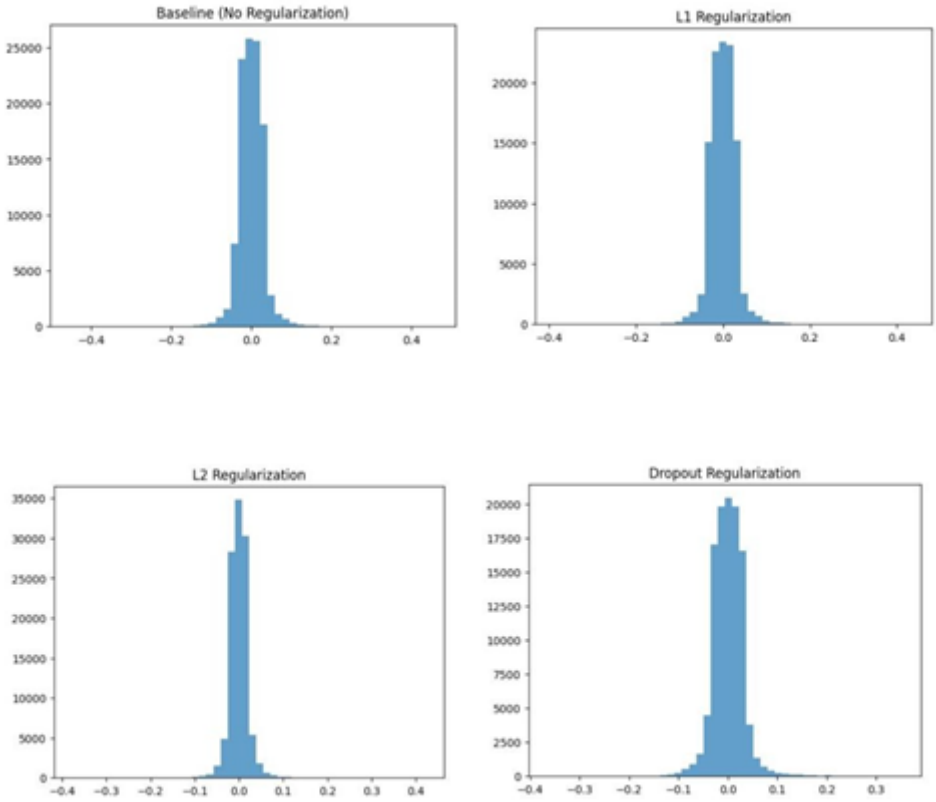


Fig.3 Weight Distribution Histogram

- **Baseline:** Wide spread of weights indicating less constraint
- **L2:** Concentrated distribution around zero
- **L1:** Sharp peak at zero, indicating sparsity
- **Dropout:** Balanced distribution

Key Insight:

- L1 eliminates unnecessary connections
- L2 controls magnitude
- Dropout improves robustness

Heatmap Analysis of Weights

Heatmaps are used to visualize learned weights across different layers.

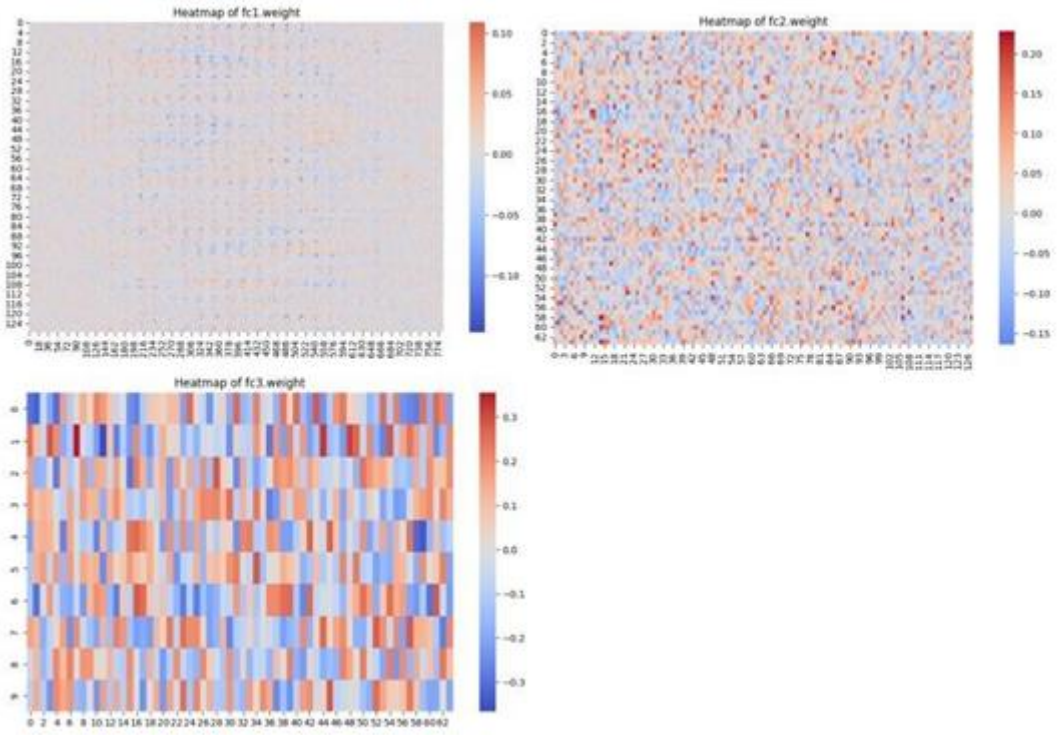
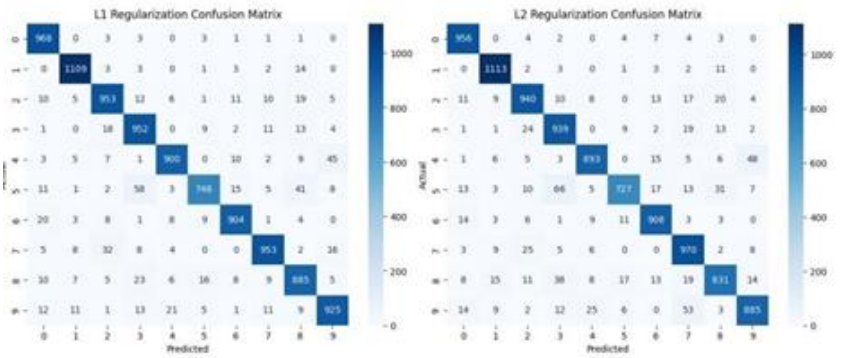


Fig.4 Weight Heatmap Visualization.

- L2 produces smoother and uniform patterns
- L1 shows sparse regions with near-zero weights
- Baseline displays irregular patterns
- Dropout shows distributed activation patterns

Confusion Matrix Analysis

Confusion matrices are used to evaluate classification performance across digit classes.



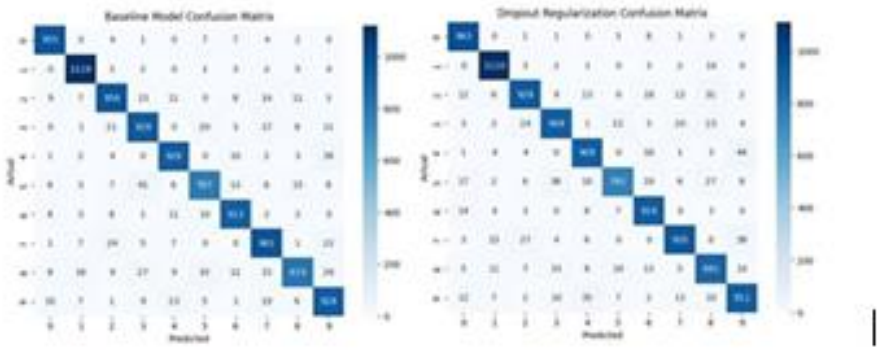


Fig.5 Confusion Matrix.

- Most predictions lie along the diagonal, indicating correct classification
- Misclassifications occur in visually similar digits (e.g., 3 vs 5, 7 vs 9)
- L1 model shows slightly better class-wise consistency

Table 2 Confusion Matrix

Test Case ID	Feature Tested	Expected Outcome	Status
TC-01	Data Loading	Correct tensor format	Success
TC-02	Training	Loss decreases	Success
TC-03	Regularization	Improved generalization	Success
TC-04	Prediction	Accurate classification	Success

DISCUSSION

The experimental results demonstrate that regularization techniques significantly influence both performance and internal behavior of neural networks. While L1 regularization promotes sparsity and achieves the best accuracy, L2 regularization ensures stability by limiting weight magnitude. Dropout enhances robustness by reducing dependency among neurons.

Unlike traditional studies that focus only on accuracy, this work highlights the importance of analyzing **weight distributions** to understand model behavior. Such analysis provides deeper insights into interpretability, efficiency, and generalization capability.

V. CONCLUSION

This study presented a comprehensive analysis of the impact of different regularization techniques on neural network performance and weight distribution. A fully connected neural network was trained on the MNIST dataset using baseline, L1 regularization, L2 regularization, and dropout approaches.

The experimental results indicate that L1 regularization achieves the highest classification accuracy by promoting sparsity in the weight parameters, thereby eliminating redundant connections. L2 regularization effectively controls weight magnitude and improves stability, although it slightly reduces accuracy due to stronger penalization. Dropout regularization enhances model robustness by reducing neuron co-adaptation, resulting in stable and reliable performance.

In addition to performance evaluation, this work emphasizes the importance of analyzing internal model characteristics such as weight distributions. The use of histograms and heatmaps provides deeper insights into how regularization influences learning behavior and network structure.

Overall, the study demonstrates that regularization plays a critical role not only in improving generalization but also in shaping the internal dynamics of neural networks. These findings contribute to the development of more efficient, interpretable, and robust machine learning models.

VI. ACKNOWLEDGEMENT

The authors would like to express their sincere gratitude to Dr. Ramya B.N. for her valuable guidance and unwavering support throughout the course of this work. Her insightful feedback and expert advice played a crucial role in shaping the direction of this research. The encouragement she provided at every stage of this work was truly invaluable. The authors are deeply thankful for her time, dedication, and commitment to their academic growth.

VII. REFERENCES

1. Goodfellow, I., Bengio, Y., Courville, A. (2016). Deep Learning. MIT Press.
2. Bishop, C. M. (2006). Pattern Recognition and Machine Learning. Springer.
3. LeCun, Y., Bottou, L., Bengio, Y., Haffner, P. (1998). Gradient-based learning applied to document recognition.
4. Chollet, F. (2023). Keras Documentation.
5. Harris, C. R., et al. (2020). Array Programming with NumPy.