Evaluating the Impact of Data Augmentation and Regularization On Image Denoising Performance Using Deep Neural Networks

Joshua Glaspey [1] **Alexander Green** [1] **Arthur Radulescu** [2]

Abstract

Image denoising plays an important role in areas like medical imaging, surveillance, and photography, where recovering clean images from noisy inputs is critical. In this project, we look at how different data augmentation and regularization techniques affect the performance of Convolutional Neural Networks (CNNs) on image denoising tasks. While these methods are commonly used in deep learning to improve generalization, their specific impact on denoising has not been studied thoroughly. Augmentation can increase the diversity of training examples, helping the model generalize across noise patterns. Regularization can reduce overfitting and encourage smoother, more stable reconstructions. Because denoising is a low-level vision task with different objectives than traditional classification, it's important to evaluate these techniques independently to understand their true contribution. We trained a baseline CNN denoising model and then evaluated it across a wide range of individual augmentations (such as brightness shifts, Gaussian noise, cropping, flipping, etc.) and regularization methods (including L1/L2 penalties, dropout, and early stopping). Our experiments use the datasets MNIST, CIFAR-10, CIFAR-100, and STL-10, and we measured results using standard metrics. Our findings show that certain augmentation techniques like Cutout and Gaussian Noise consistently outperform others, while Base (no augmentation) often provides strong baseline performance across different noise levels.

[1]Department of Electrical and Computer Engineering, University of Central Florida, Orlando, United States of America [2]Department of Electrical Engineering, University of Central Florida, Orlando, United States of America. Correspondence to: Joshua Glaspey <al101178@ucf.edu>, Alexander Green <jo248954@ucf.edu>, Arthur Radulescu <ar052857@ucf.edu>.

Proceedings of the 41^{st} International Conference on Machine Learning, Vienna, Austria. PMLR 235, 2024.

1. Introduction

Image denoising remains a key challenge in computer vision, with applications in photography, medical imaging, and satellite analysis. The task involves reconstructing clean images from noisy inputs, which becomes increasingly difficult across diverse noise patterns and datasets. While CNNs have become the standard approach due to their capacity to learn rich image priors, model performance depends not just on architecture but also on how the network is trained.

Most existing work emphasizes architectural innovation—such as the use of depthwise separable convolutions in DenoisingNet to improve efficiency (1)—but comparatively little attention has been given to training strategies like data augmentation and regularization. Augmentation is commonly used to expand dataset diversity and improve generalization, particularly when data is limited. Surveys by Khosla et al. and Sengupta et al. outline a broad set of augmentation techniques, from geometric transformations to color and noise adjustments (2; 3). However, in pixel-wise tasks like denoising, not all transformations preserve the clean-noisy correspondence needed for effective learning.

Regularization techniques, including L1/L2 penalties, dropout, and early stopping, are widely used to reduce overfitting and encourage stability during training. While these have been extensively studied in classification settings, their specific impact on low-level vision tasks like denoising is less clear. As Moradi et al. point out, regularization effects can vary depending on model size, dataset scale, and training constraints (4).

This project aims to fill that gap by systematically evaluating how a range of individual data augmentation and regularization strategies affect CNN-based image denoising. We train a baseline model on four datasets—MNIST, CIFAR-10, CIFAR-100, and STL-10—and test augmentations including flipping, cropping, brightness, contrast, Gaussian noise, and a custom method proposed by Sengupta et al. (3). We also examine L1, L2, dropout, and early stopping individually. By isolating each method and measuring both quantitative and qualitative outcomes, we provide a clearer understanding of how these commonly used training strategies influence denoising performance.

2. Related Works

2.1. Image Denoising with CNNs

Image denoising using convolutional neural networks (CNNs) has become a dominant approach in low-level vision tasks. CNN-based models operate by learning a mapping function f_θ that transforms a noisy input image x into a clean estimate $\hat{y}$, with the goal of minimizing a loss function such as the mean squared error (MSE):

$$L = E_{x,y}[\|f_\theta(x) - y\|_2^2]$$

where y is the ground truth clean image. Many CNN denoising architectures use deep residual learning to model the noise component instead of the clean image directly, accelerating convergence and improving accuracy. Recent trends include the use of separable convolutions, attention mechanisms, and multi-scale representations to better capture image structure and noise patterns (5). These models tend to be evaluated on benchmark datasets with fixed noise levels and relatively controlled input conditions. While the architectural focus has driven notable improvements in image quality metrics (e.g. PSNR and SSIM), fewer studies have examined how training strategies-like data augmentation or regularization-can systematically affect denoising outcomes across datasets and noise conditions.

2.2. Data Augmentation for Deep Learning

Data augmentation is a standard practice in deep learning, especially useful in scenarios with limited data or when improving robustness to input variation. It operates by transforming the input data x via a distribution-preserving function $T(x) \sim T$, effectively enriching the training set and encouraging models to generalize across unseen examples. Shorten and Khoshgoftaar (6) categorize augmentation strategies into geometric transformations (e.g. flipping, cropping, rotation), photometric changes (e.g. brightness, color jitter), and data synthesis techniques using generative models. These methods can reduce overfitting by preventing the model from memorizing spurious correlations in the training data. Khosla and Saini (2) outline how augmentations expand the support of the input distribution $P(X)$, which can lower generalization error. Sengupta et al. (3) further suggests that chaining multiple augmentations in a specific sequence can boost performance more than using them individually. However, most studies focus on classification tasks, where label invariance is easier to preserve. In denoising, the requirement for strict pixel-to-pixel alignment between noisy and clean image pairs limits the applicability of some transformations. This makes it unclear which augmentations are beneficial, harmful, or neutral when used in low-level restoration tasks.

2.3. Regularization Techniques in Deep Models

Regularization strategies are designed to improve generalization by controlling model complexity or introducing constraints during training. Common approaches include norm-based penalties such as L1 and L2 regularization, where terms like $\lambda\|\omega\|_1$ or $\lambda\|\omega\|_2^2$ are added to the loss function to penalize large or sparse weights (4). Dropout introduces stochasticity during training by randomly deactivating neurons, encouraging the model to learn redundant and robust representations (8). Batch normalization, early stopping, and noise injection are also widely used, each operating at different stages of the training pipeline. Kukačka et al. (7) categorizes these methods based on whether they modify the input space, the architecture, or the optimization path. While extensively studied in the context of classification, regularization's effects in pixel-wise regression tasks like denoising are not well understood. The constraints these methods introduce may interfere with the model's capacity to recover fine spatial detail.

2.4. Current Limitations of Prior Work When Applied to Denoising

Although data augmentation and regularization have demonstrated strong utility in high-level vision tasks, their role in image denoising remains underexplored and potentially domain-sensitive. Shorten and Khoshgoftaar (6) notes that many augmentations are not inherently label-preserving in low-level tasks. For example, a simple horizontal flip might preserve a classification label but misalign the noise-clean image mapping necessary for denoising. Similarly, augmentations that involve cropping, rotation, or color space manipulation may alter the pixel-level semantics needed for accurate reconstruction. Balestriero et al. (9) demonstrates that the impact of these methods varies across tasks and datasets, often in unpredictable ways. Hernandez-Garcia and König (10) suggests that in some cases, augmentation alone can act as an implicit form of regularization. However, this conclusion is based on classification tasks and may not hold when the optimization target involves pixel recovery. As a result, there is a demand to evaluate these techniques specifically within the denoising domain. This project addresses that gap by systematically testing both augmentation and regularization strategies-individually and in combination-on a baseline CNN trained for image denoising across multiple datasets and noise settings.

3. Methodology

3.1. Augmentation and Regularization Motivation

The selected augmentation strategies are: brightness adjustment, color jitter, contrast modification, cutout/erasing, flipping, Gaussian noise, random cropping, rotation, scaling,

and shearing. Additionally, the regularization techniques are: L1 regularization, L2 regularization, dropout, and early stopping.

The motivation for the selection of these augmentation techniques is that they enhance the performance of denoising neural networks. These techniques prevent overfitting by creating diverse variations of training data, allowing the network to learn generalizable features rather than memorizing specific noise patterns. They also improve the model's robustness by emulating various real-world conditions and degradations. When clean-noisy image pairs are limited, augmentations effectively multiply the available training samples, improving data efficiency while providing a regularization effect that strengthens the network's ability to handle corrupted inputs. These benefits lead to denoising models that generalize better.

Our motivation for selecting the previously mentioned regularization techniques is to prevent overfitting and improve generalization. Neural networks, particularly when trained on limited datasets, tend to memorize noise patterns specific to the training data rather than learning to distinguish universal noise characteristics. Each regularization method addresses this challenge through different mechanisms: L1 and L2 regularization control weight magnitudes but with different effects on sparsity, dropout creates robustness by preventing co-adaptation of neurons, and early stopping prevents the model from fitting noise in the training data.

3.2. Experimental Procedure and Datasets

To evaluate the isolated effects of data augmentation and regularization techniques on image denoising performance, we constructed a controlled experimental framework that systematically applies each method independently. For every configuration, a baseline CNN was trained on noisy-clean image pairs, with Gaussian noise artificially randomly introduced. We tested three distinct noise levels-15% (low), 25% (medium), and 50% (high)-to simulate a range of realistic conditions from mild sensor noise to heavily degraded inputs. Each augmentation and regularization strategy was tested separately using identical training conditions, allowing for fair comparison across methods. For each regularization, three separate values were tested to account for hyperparameter tuning and discover the optimal performance per dataset. Model performance was tracked using training and testing accuracy and loss, and computation time. The CNN model architecture is in Section 7.4. (Table 19)

We selected four datasets to capture a wide range of visual characteristics and complexity levels: MNIST, CIFAR-10, CIFAR-100, and STL-10. MNIST provides simple grayscale images of handwritten digits, serving as a low-resolution benchmark for structured, low-variability data. CIFAR-10 and CIFAR-100 consist of small color images containing natural objects, with CIFAR-100 offering increased intra-class diversity due to its larger label set. STL-10 includes higher-resolution images of natural scenes and objects, creating a more challenging setting for spatial reconstruction tasks. This allows us to evaluate how augmentation and regularization strategies generalize across different domains.

3.3. Base Experiment

The base experiment established a controlled benchmark for evaluating the neural network's denoising performance without the use of regularization or data augmentation techniques. The neural network model was trained on noisy image inputs created from clean datasets such as MNIST, CIFAR10, CIFAR100, and STL10. Each was subjected to varying levels of Gaussian noise. No additional regularization strategies were applied, and no data augmentation techniques were used to artificially expand or alter the training dataset.

The image processing pipeline for the base experiment followed a consistent structure. First, the selected dataset was imported in its clean form. Gaussian noise are then added to each image, resulting in a noisy version that serves as the network's input. These noisy images are passed through the neural network, which was trained to denoise the images. The model's performance was evaluated by comparing its denoised outputs to the original clean images.

By isolating the model from any auxiliary training enhancements, the base experiment provided a reliable reference point. This allowed for clear, quantitative comparisons when evaluating the impact of individual data augmentation and regularization techniques in subsequent experiments.

4. Augmentation Experimental Results

4.1. Noise Level 15%

Method	Test Acc	Test Loss	Time (s)
Base	0.96	0.0014	578
Brightness	0.95	0.0022	574
Color Jitter	0.96	0.0016	580
Contrast	0.95	0.0022	576
Custom Aug. 1	0.86	0.0126	603
Cutout	0.96	0.0015	575
Flipping	0.71	0.0754	572
Gaussian Noise	0.96	0.0015	571
Random Crop	0.72	0.0507	572
Rotation	0.89	0.0078	598
Scaling	0.95	0.0019	597
Shearing	0.92	0.0061	599

Table 1. MNIST Dataset Results

Method	Test Acc	Test Loss	Time (s)
Base	0.95	0.0023	635
Brightness	0.82	0.0063	632
Color Jitter	0.87	0.0044	699
Contrast	0.85	0.0050	634
Custom Aug. 1	0.63	0.0147	654
Cutout	0.95	0.0024	636
Flipping	0.39	0.0384	630
Gaussian Noise	0.94	0.0025	622
Random Crop	0.68	0.0163	622
Rotation	0.85	0.0054	649
Scaling	0.93	0.0028	651
Shearing	0.89	0.0045	650

Table 2. CIFAR-10 Dataset Results

Method	Test Acc	Test Loss	Time (s)
Base	0.95	0.0023	618
Brightness	0.79	0.0068	629
Color Jitter	0.90	0.0039	700
Contrast	0.88	0.0043	634
Custom Aug. 1	0.62	0.0141	686
Cutout	0.95	0.0023	637
Flipping	0.39	0.0379	631
Gaussian Noise	0.94	0.0025	621
Random Crop	0.73	0.0129	623
Rotation	0.86	0.0050	679
Scaling	0.92	0.0032	679
Shearing	0.88	0.0049	681

Table 3. CIFAR-100 Dataset Results

Method	Test Acc	Test Loss	Time (s)
Base	0.95	0.0022	437
Brightness	0.67	0.0125	435
Color Jitter	0.90	0.0038	567
Contrast	0.90	0.0038	430
Custom Aug. 1	0.58	0.0181	534
Cutout	0.94	0.0026	434
Flipping	0.31	0.0472	429
Gaussian Noise	0.94	0.0024	429
Random Crop	0.56	0.0245	429
Rotation	0.65	0.0167	590
Scaling	0.76	0.0122	529
Shearing	0.81	0.0079	546

Table 4. STL-10 Dataset Results

At the 15% noise level, the augmentation experiments revealed consistent trends across all datasets. Of the data augmentation techniques evaluated, the Base (no augmentation), Cutout, and Gaussian Noise configurations consistently achieved the highest test accuracies and lowest test losses. This suggests that these techniques are either effectively preserving important image features or enhancing the model's robustness to noise, particularly in the low-noise regime. Cutout and Gaussian Noise, in particular, may help the network generalize by introducing structured occlusions or additional variability, while the Base case benefits from the simplicity of direct noise-to-clean image training.

Computation time was generally uniform across augmentation methods, showing no statistically significant variation. Nonetheless, Flipping, Gaussian Noise, and Random Crop emerged as the most computationally efficient, consistently recording the shortest runtimes. Despite their speed, however, Flipping and Random Crop were the least effective in terms of accuracy and loss, with both techniques consistently ranking lowest across datasets. This performance gap indicates that, while fast, these augmentations may introduce spatial transformations or distortions that hinder the network's ability to accurately reconstruct clean images. Consequently, while speed is desirable, augmentation methods like Cutout and Gaussian Noise offer a better trade-off between efficiency and denoising performance at this noise level.

4.2. Noise Level 25%

Method	Test Acc	Test Loss	Time (s)
Base	0.94	0.0029	570
Brightness	0.92	0.0047	575
Color Jitter	0.94	0.0030	581
Contrast	0.94	0.0035	576
Custom Aug. 1	0.84	0.0149	602
Cutout	0.94	0.0030	577
Flipping	0.70	0.0730	573
Gaussian Noise	0.94	0.0030	573
Random Crop	0.71	0.0568	573
Rotation	0.88	0.0099	599
Scaling	0.93	0.0035	599
Shearing	0.89	0.0090	599

Table 5. MNIST Dataset Results

At the 25% noise level, the augmentation results closely mirrored those observed at 15%. Across all datasets, the Base, Cutout, and Gaussian Noise augmentations continued to demonstrate superior performance, consistently yielding the highest accuracies and lowest losses. Flipping and Random Crop again ranked lowest in accuracy and highest in loss, reaffirming their limited effectiveness for denoising tasks at this noise level.

Method	Test Acc	Test Loss	Time (s)
Base	0.88	0.0042	775
Brightness	0.80	0.0068	795
Color Jitter	0.83	0.0058	980
Contrast	0.82	0.0062	814
Custom Aug. 1	0.54	0.0200	680
Cutout	0.88	0.0043	666
Flipping	0.37	0.0410	661
Gaussian Noise	0.88	0.0043	635
Random Crop	0.66	0.0163	636
Rotation	0.80	0.0072	675
Scaling	0.87	0.0046	676
Shearing	0.82	0.0070	672

Table 6. CIFAR-10 Dataset Results

Method	Test Acc	Test Loss	Time (s)
Base	0.88	0.0042	640
Brightness	0.80	0.0065	658
Color Jitter	0.83	0.0056	720
Contrast	0.85	0.0053	634
Custom Aug. 1	0.61	0.0148	678
Cutout	0.88	0.0043	637
Flipping	0.40	0.0348	641
Gaussian Noise	0.88	0.0043	643
Random Crop	0.68	0.0159	623
Rotation	0.83	0.0062	668
Scaling	0.87	0.0046	654
Shearing	0.82	0.0069	657

Table 7. CIFAR-100 Dataset Results

Method	Test Acc	Test Loss	Time (s)
Base	0.90	0.0037	447
Brightness	0.57	0.0143	441
Color Jitter	0.87	0.0049	596
Contrast	0.69	0.0103	522
Custom Aug. 1	0.51	0.0236	528
Cutout	0.89	0.0041	533
Flipping	0.30	0.0488	525
Gaussian Noise	0.90	0.0039	526
Random Crop	0.62	0.0207	526
Rotation	0.61	0.0190	529
Scaling	0.75	0.0128	528
Shearing	0.78	0.0092	529

Table 8. STL-10 Dataset Results

Computation time remained largely consistent across methods, with no meaningful differences in runtime. However, Flipping, Gaussian Noise, and Random Crop maintained their position as the fastest augmentations, despite the clear trade-off in denoising quality for Flipping and Random Crop. These results further emphasize the effectiveness of Cutout and Gaussian Noise as reliable augmentations for moderate noise environments.

4.3. Noise Level 50%

Method	Test Acc	Test Loss	Time (s)
Base	0.90	0.0085	592
Brightness	0.90	0.0096	597
Color Jitter	0.90	0.0091	597
Contrast	0.90	0.0092	563
Custom Aug. 1	0.78	0.0243	599
Cutout	0.90	0.0086	561
Flipping	0.69	0.0694	557
Gaussian Noise	0.90	0.0086	557
Random Crop	0.67	0.0582	557
Rotation	0.83	0.0166	580
Scaling	0.89	0.0092	578
Shearing	0.85	0.0174	581

Table 9. MNIST Dataset Results

Method	Test Acc	Test Loss	Time (s)
Base	0.75	0.0091	616
Brightness	0.64	0.0126	630
Color Jitter	0.72	0.0101	703
Contrast	0.70	0.0111	634
Custom Aug. 1	0.56	0.0186	643
Cutout	0.74	0.0092	637
Flipping	0.35	0.0403	625
Gaussian Noise	0.74	0.0094	611
Random Crop	0.63	0.0176	614
Rotation	0.71	0.0107	640
Scaling	0.74	0.0095	654
Shearing	0.72	0.0105	639

Table 10. CIFAR-10 Dataset Results

At the 50% noise level, the results remained consistent with those at 15% and 25%. Base, Cutout, and Gaussian Noise continued to deliver the best performance across all datasets, while Flipping and Random Crop consistently performed the worst in terms of accuracy and loss.

Computation times showed no significant variation, with Flipping, Gaussian Noise, and Random Crop remaining the fastest. Overall, Cutout and Gaussian Noise again proved to be the most effective augmentation techniques, balancing both efficiency and denoising performance even under high noise conditions.

Method	Test Acc	Test Loss	Time (s)
Base	0.75	0.0092	1164
Brightness	0.65	0.0128	1135
Color Jitter	0.69	0.0113	1064
Contrast	0.72	0.0101	997
Custom Aug. 1	0.55	0.0192	1019
Cutout	0.75	0.0092	1000
Flipping	0.36	0.0380	994
Gaussian Noise	0.74	0.0093	982
Random Crop	0.64	0.0166	983
Rotation	0.70	0.0113	1014
Scaling	0.74	0.0097	1016
Shearing	0.72	0.0108	1014

Table 11. CIFAR-100 Dataset Results

Method	Test Acc	Test Loss	Time (s)
Base	0.80	0.0075	464
Brightness	0.58	0.0165	468
Color Jitter	0.75	0.0093	650
Contrast	0.77	0.0087	500
Custom Aug. 1	0.46	0.0265	479
Cutout	0.79	0.0079	494
Flipping	0.31	0.0494	475
Gaussian Noise	0.78	0.0083	472
Random Crop	0.59	0.0214	473
Rotation	0.60	0.0194	475
Scaling	0.70	0.0135	482
Shearing	0.75	0.0105	478

Table 12. STL-10 Dataset Results

4.4. Conclusions on Findings

Our analysis across three noise levels consistently showed that Base, Cutout, and Gaussian Noise augmentations achieved superior performance in image denoising tasks. Notably, at the highest noise level (50%), both Cutout and Gaussian Noise occasionally outperformed the Base configuration, particularly in the CIFAR-100 and MNIST datasets. This suggests these augmentation techniques become increasingly valuable as noise severity increases.

4.4.1. Base

The strong performance of Base configurations establishes an important baseline, but doesn't imply augmentation should be avoided. Rather, it suggests augmentations should be strategically applied to expand your training data, creating artificial variations that enhance model robustness without fundamentally altering core image characteristics needed for denoising.

4.4.2. Cutout

Cutout augmentation randomly masks image regions, forcing the network to develop context-based reconstruction capabilities. This technique is particularly effective for high-noise environments where the network must learn to recover information from context rather than relying solely on direct pixel values. This teaches the model to infer missing information from surrounding pixels—a valuable skill for denoising heavily corrupted regions. Its effectiveness at 50% noise indicates its value for applications dealing with significantly degraded images.

4.4.3. Gaussian Noise

Adding controlled Gaussian noise during training creates a form of "noise inoculation," helping the network distinguish between signal and noise patterns. This technique improves generalization by teaching the model to identify and preserve important features amid varying noise conditions. This approach is ideal when denoising performance must generalize across varying noise types and intensities.

4.4.4. Flipping and Random Crop

The poor performance of Flipping and Random Crop likely stems from their disruption of spatial relationships crucial for denoising. These transformations can introduce unnatural artifacts that hinder the network's ability to learn meaningful noise patterns.

5. Regularization Experimental Results

5.1. L1 Regularization

We experimented with L1 values of $1e^{-5}$, $1e^{-4}$, and $1e^{-3}$. Most tests indicated that $1e^{-5}$ yielded the best performance across datasets and noise levels, with little variation observed. The following table shows results using $L1 = 1e^{-5}$.

Dataset	Noise %	Test Acc	Test Loss	Time (s)
MNIST	15%	0.95	0.0019	624
	25%	0.93	0.0037	618
	50%	0.89	0.0106	618
CIFAR-10	15%	0.89	0.0041	667
	25%	0.83	0.0060	643
	50%	0.68	0.0116	667
CIFAR-100	15%	0.87	0.0048	670
	25%	0.81	0.0066	646
	50%	0.67	0.0124	643
STL-10	15%	0.86	0.0054	424
	25%	0.74	0.0101	427
	50%	0.75	0.0096	426

Table 13. L1 Regularization Results Across Different Datasets and Noise Levels

5.2. L2 Regularization

L2 values of $1e^{-4}$, $1e^{-3}$, and $1e^{-2}$ were tested. Across all runs, $1e^{-4}$ consistently performed best and showed minimal deviation. The table below reflects performance with $L2 = 1e^{-4}$.

Dataset	Noise %	Test Acc	Test Loss	Time (s)
MNIST	15%	0.95	0.0019	636
	25%	0.93	0.0036	629
	50%	0.88	0.0110	630
CIFAR-10	15%	0.92	0.0033	678
	25%	0.85	0.0053	653
	50%	0.71	0.0107	654
CIFAR-100	15%	0.91	0.0036	679
	25%	0.83	0.0058	651
	50%	0.71	0.0107	654
STL-10	15%	0.88	0.0046	424
	25%	0.86	0.0053	426
	50%	0.76	0.0094	426

Table 14. L2 Regularization Results Across Different Datasets and Noise Levels

5.3. Dropout

Dropout rates of $0.2, 0.3, and 0.4$ were evaluated. A dropout rate of 0.2 was the most effective and consistent across trials. Results below use $DR = 0.2$.

Dataset	Noise %	Test Acc	Test Loss	Time (s)
MNIST	15%	0.96	0.0015	589
	25%	0.94	0.0030	580
	50%	0.90	0.0086	582
CIFAR-10	15%	0.94	0.0027	643
	25%	0.87	0.0048	657
	50%	0.73	0.0095	620
CIFAR-100	15%	0.93	0.0031	635
	25%	0.87	0.0048	657
	50%	0.73	0.0095	620
STL-10	15%	0.93	0.0030	433
	25%	0.89	0.0044	435
	50%	0.78	0.0082	435

Table 15. Dropout Results Across Different Datasets and Noise Levels

5.4. Early Stopping

We evaluated patience values of $5, 10, and 15$ for early stopping. While results varied slightly across runs, a patience value of 10 was optimal most frequently. The table below reflects results using $ES = 10$.

Dataset	Noise %	Test Acc	Test Loss	Time (s)
MNIST	15%	0.96	0.0014	527
	25%	0.94	0.0029	931
	50%	0.90	0.0085	410
CIFAR-10	15%	0.95	0.0023	714
	25%	0.89	0.0042	602
	50%	0.75	0.0091	604
CIFAR-100	15%	0.95	0.0023	643
	25%	0.88	0.0042	591
	50%	0.75	0.0091	604
STL-10	15%	0.95	0.0022	420
	25%	0.90	0.0037	421
	50%	0.80	0.0077	422

Table 16. Early Stopping Results Across Different Datasets and Noise Levels

5.5. Conclusions and Findings

5.5.1. L1 Regularization

The optimal L1 regularization value was found to be $1e$-5. This value applies a minimal penalty to the absolute magnitude of weights, which is suitable for image denoising tasks where overly sparse representations may discard subtle yet meaningful pixel-level features. Lower L1 regularization preserves more model flexibility, enabling the network to capture finer image details needed to distinguish noise from structure.

5.5.2. L2 Regularization

The best-performing L2 value was $1e$-4. This level of weight decay provides sufficient constraint to reduce overfitting while still allowing the model to learn complex image features. For image denoising, where generalization across varying noise patterns is essential, this moderate regularization encourages smoother weight distributions without overly restricting the model's capacity.

5.5.3. Dropout

A dropout rate of 0.2 was most effective. This relatively low rate prevents excessive neuron deactivation, which is critical in image denoising where spatial coherence and feature continuity must be preserved. This level of dropout introduces beneficial stochasticity during training, improving robustness without severely impairing the network's ability to learn clean image representations.

5.5.4. Early Stopping

The optimal early stopping patience was found to be 10 epochs. This strikes a balance between allowing the model to continue improving and halting before overfitting to noisy training data. For image denoising networks, where the risk

of overfitting to noise is high, early stopping effectively limits unnecessary parameter updates once validation performance begins to degrade.

5.5.5. Overall Comparison

Among all methods tested, early stopping delivered the best overall performance across datasets and noise levels. This result is expected for image denoising deep neural networks, as early stopping directly monitors validation loss and halts training before the network begins to memorize noisy patterns. This adaptive control aligns well with the denoising objective of generalizing to clean image structure across varying noise conditions.

6. Conclusion

Our systematic investigation into data augmentation and regularization techniques for image denoising neural networks revealed several key findings. Base (no augmentation), Cutout, and Gaussian Noise consistently outperformed other augmentation methods across varying noise levels and datasets. These techniques preserve critical spatial relationships within images while introducing beneficial variability that enhances model robustness. Spatial transformations like Flipping and Random Crop performed poorly because they disrupt pixel-level correspondences essential for denoising tasks. Unlike classification problems where label preservation is sufficient, denoising requires maintaining precise spatial alignment between noisy and clean image pairs, which explains why certain augmentations prove detrimental despite their success in other computer vision tasks.

Among regularization techniques, Early Stopping demonstrated superior performance, particularly at higher noise levels. This method's effectiveness stems from its adaptive approach to optimization, preventing the network from overfitting to noise patterns by monitoring validation performance and halting training before memorization occurs. Dropout (at 0.2) also performed well by introducing stochasticity without excessively compromising spatial continuity. L2 regularization at $1e^{-4}$ provided an effective balance between constraint and flexibility, whereas L1's optimal value ($1e^{-5}$) was significantly lower, indicating that excessive sparsity can hamper denoising by discarding subtle yet important features.

These findings highlight the importance of task-specific optimization strategies for image denoising. The success of Cutout suggests that forcing networks to infer missing information from surrounding context enhances robustness to corrupted pixels. Similarly, Gaussian Noise augmentation's effectiveness demonstrates that exposing models to varied noise patterns during training creates a form of "noise inoculation" that improves generalization. Both techniques complement the denoising objective by specifically targeting the network's ability to distinguish signal from noise rather than introducing arbitrary variations that might be beneficial for classification but counterproductive for pixel-wise reconstruction tasks.

Our results provide practical guidance for implementing deep learning-based image denoising systems across various application domains. The consistent performance patterns across different datasets (from simple MNIST to complex STL-10) and noise levels (15%, 25%, and 50%) suggest that these findings are generalizable. Future work should explore combining the best-performing methods, particularly Early Stopping with Cutout or Gaussian Noise augmentation, as their complementary mechanisms may yield further improvements in denoising performance while maintaining computational efficiency.

References

[1] DenoisingNet: an efficient convolutional neural network for image denoising. IEEE Conference Publication, 2019.

[2] Khosla, et al. *Enhancing Performance of Deep Learning Models with different Data Augmentation Techniques: A Survey*. IEEE Conference Publication, 2020.

[3] Sengupta, et al. *Enhancing Performance of Deep Learning Models with a Novel Data Augmentation Approach*. IEEE Conference Publication, 2023.

[4] Moradi, R., Berangi, R., & Minaei, B. *A survey of regularization strategies for deep models*. Artificial Intelligence Review, 53(6), 3947–3986, 2019.

[5] *Image Denoising using Convolutional Neural Network*. IEEE Conference Publication, 2022.

[6] Shorten, C., & Khoshgoftaar, T. M. *A survey on Image Data Augmentation for Deep Learning*. Journal of Big Data, 6(1), 2019.

[7] Kukačka, J., Golkov, V., & Cremers, D. *Regularization for Deep Learning: A Taxonomy*. arXiv.org, 2017.

[8] Murugan, P., & Durairaj, S. *Regularization and Optimization strategies in Deep Convolutional Neural Network*. arXiv.org, 2017.

[9] Balestriero, R., Bottou, L., & LeCun, Y. *The Effects of Regularization and Data Augmentation are Class Dependent*. Proceedings of NeurIPS, 2022.

[10] Hernández-García, A., & König, P. *Data augmentation instead of explicit regularization*. arXiv.org, 2018.

7. Supplimentary Section

7.1. Augmentation Background and Motivation

7.1.1. Brightness

Brightness augmentation modifies the overall luminance of training images by adding or subtracting a constant value to all pixels. This technique helps models become robust to lighting variations and exposure differences commonly found in real-world imaging scenarios. For denoising tasks, it can be particularly relevant when dealing with images captured under varying lighting. Our implementation used a brightness factor of 1.5, providing reasonable variation without introducing extreme artifacts that could confuse the network's training process.

$$I_{\text{bright}}(\alpha, \beta) = I(\alpha, \beta) + \Delta b$$

7.1.2. Color Jitter

Color jitter adjusts an image's color at random, doing so by adding a value to the pixel values of all pixels in an image. This comprehensive approach simulates natural color variations encountered in real-world scenarios and provides robustness against a wide range of color perturbations. For color datasets (CIFAR-10, CIFAR-100, STL-10), this augmentation helps ensure the model doesn't overly rely on specific color patterns for recognizing structures during denoising. Our implementation used moderate jitter factors (brightness of 0.2, contrast of 0.2, saturation of 0.2, hue of 0.1) to preserve the overall image semantics while introducing sufficient variation.

$$I_{\text{jitter}}(\alpha, \beta) = (I(\alpha, \beta) + \Delta c)$$

7.1.3. Contrast

Contrast augmentation adjusts the contract of an image by multiplying pixel values by a scaling factor. By modifying the intensity range, this technique helps models learn features that are unaffected by contrast variations. In denoising applications, contrast augmentation is particularly useful for handling images from different imaging devices or processing pipelines that may produce different contrast levels. We implemented contrast adjustments using a factor of 1.5 to introduce meaningful variation without compromising image quality or basic structure recognition.

$$I_{\text{contrast}}(\alpha, \beta) = (I(\alpha, \beta) - 0.5) \cdot \Delta c + 0.5$$

7.1.4. Cutout

Cutout randomly masks rectangular regions of the input image with zeros or random noise. Unlike dropout which operates at the feature level, cutout works directly on the input, forcing the network to use surrounding context to recover missing information. This technique has shown success in classification tasks by preventing overfitting to specific image regions. For denoising, cutout may help the network develop stronger contextual reasoning capabilities, which could improve reconstruction of heavily corrupted areas. Our implementation used square cutouts with sides equal to 20.

7.1.5. Flipping

Flipping (horizontal or vertical) is among the most common augmentations in computer vision. For classification tasks, flipping preserves class identity while adding variation. However, for pixel-level tasks like denoising, flipping introduces a fundamental challenge: it alters the spatial relationship between noise patterns and underlying image structures. We included flipping to evaluate the model's sensitivity to such misalignment and determine whether denoising performance degrades in the presence of spatial inversion. We implemented a sequence of horizontal and vertical flipping to test whether the denoising task could benefit from this additional variation despite the potential misalignment issues in noise-clean correspondences.

$$f(\omega_x, \omega_y) = \begin{bmatrix} -1 & 0 \\ 0 & 1 \end{bmatrix}$$

7.1.6. Gaussian Noise

Gaussian noise augmentation adds random noise drawn from a Gaussian distribution to each pixel in an image. This technique creates additional noisy samples beyond the base noise level, effectively teaching the network to handle varying noise intensities and patterns. For denoising tasks, this form of noise immunization can be particularly valuable as it directly addresses the task objective. We implemented this with controlled standard deviation (0.1) to ensure the additional noise provided a meaningful training stimulus without overwhelming the original image structures.

$$N_{\text{noise}}(\alpha, \beta) = I(\alpha, \beta) + \varepsilon$$

7.1.7. Random Crop

Random crop extracts portions of images at training time, effectively focusing the network on different regions while maintaining the original resolution. This technique can help models generalize across image positions and scales. For denoising, however, cropping presents challenges similar to flipping, as it may disrupt the alignment between noisy and clean pairs. Our motivation is to assess whether restricting the visible context encourages the model to rely on more local features during reconstruction. We implemented random crops that preserved at least 80% but no more than 90% of the original image area to evaluate whether this additional variation would benefit denoising performance despite the potential alignment issues.

7.1.8. ROTATION

Rotation augmentation applies angular transformations to input images, helping models develop rotation invariance. For denoising tasks, small rotations can be valuable for handling minor orientation variations in real-world scenarios. However, like flipping and cropping, rotation may disrupt pixel-level correspondence between noisy and clean images. Therefore, our motivation follows the same suit of logic: to test the network's ability to generalize under different orientation shifts and observe its ability to restore fine image structure from moderate spatial distortion. Our implementation used moderate rotations ($+45°$) to test whether the benefits of additional training variation outweighed the potential alignment issues.

$$f(\psi) = \begin{bmatrix} \cos\psi & -\sin\psi \\ \sin\psi & \cos\psi \end{bmatrix}$$

7.1.9. SCALING

Scaling augmentation resizes images by a random factor. This technique helps models handle objects at different scales and distances. For denoising tasks, scaling can improve robustness to different resolutions and zoom levels but may introduce interpolation artifacts that affect fine detail reconstruction. We implemented scaling with factors between 0.8 and 1.2 to introduce meaningful variation while minimizing excessive distortion.

$$f(\varepsilon_x, \varepsilon_y) = \begin{bmatrix} \varepsilon_x & 0 \\ 0 & \varepsilon_y \end{bmatrix}$$

7.1.10. SHEARING

Shearing applies a transformation where points along one axis remain fixed while others are shifted parallel to that axis by a distance proportional to their perpendicular distance from it. This technique simulates perspective changes and geometric distortions. For denoising, shearing tests whether models can handle geometric transformations of noise patterns relative to underlying image structures. We implemented a shear factor of ± 20 degrees to introduce noticeable but not extreme distortion.

$$f(\omega_x, \omega_y) = \begin{bmatrix} 1 & \omega_x \\ \omega_y & 1 \end{bmatrix}$$

7.1.11. CUSTOM AUGMENTATION 1

This augmentation chain implements the sequential procedure recommended by Sengupta et al. (3), combining multiple transformations in a specific order. The sequence applies rescaling by a random scale between 0.9 and 1.1, rotating by a random angle between 0 and 20, flipping vertically with probability 0.5, and finally converting to grayscale with probability 0.5. We included this to serve as a literature-base reference policy to compare individually tested augmentations to a combined approach and observe whether multiple transformations can improve denoising performance. This approach targets comprehensive invariance while attempting to preserve most pixel-level correspondences required for denoising. The careful parameter selection aims to create variation without introducing excessive distortion that might compromise the denoising objective.

7.2. Regularization Background and Motivation

7.2.1. L1 REGULARIZATION

L1 regularization adds a penalty term proportional to the absolute values of model weights to the loss function, encouraging sparse representations by driving many weights toward zero. This approach effectively performs feature selection, retaining only the most important connections. For denoising tasks, L1 regularization may help identify and preserve the most critical features for reconstruction while discarding less relevant ones. We implemented this with a strength factor of $1e^{-5}$ to provide regularization without overly constraining the model's capacity to learn complex denoising patterns. In the equation below, w_i represents the model weights, and λ is the regularization strength.

$$\mathcal{L}_{L1} = \lambda \sum_i |w_i|$$

7.2.2. L2 REGULARIZATION

L2 regularization adds a penalty proportional to the squared magnitude of weights, preventing any single feature from dominating and encouraging weight values to remain small and diffused. Unlike L1, L2 doesn't produce sparse solutions but creates more balanced parameter distributions. For denoising tasks, L2 regularization may promote smoother reconstructions by preventing overreliance on specific features. We implemented this with a lambda value of $1e^{-4}$ to ensure meaningful regularization without excessive constraint.

$$\mathcal{L}_{L2} = \lambda \sum_i w_i^2$$

7.2.3. DROPOUT

Dropout randomly deactivates neurons during training with a specified probability, forcing the network to develop redundant representations and preventing co-adaptation of feature detectors. This technique simulates ensemble learning within a single network. For denoising tasks, dropout may improve generalization to unseen noise patterns by ensuring no single feature pathway becomes essential. We implemented a dropout rate of 0.2 in convolutional layers to provide regularization while maintaining sufficient capacity for detailed reconstruction.

7.2.4. Early Stopping

Early stopping monitors validation performance during training and halts the process when performance begins to degrade, preventing overfitting by optimizing the trade-off between underfitting and overfitting. For denoising tasks, early stopping helps identify the point where the model best generalizes to unseen noise patterns without memorizing training noise. We implemented this with patience of 10 epochs and monitored validation loss to determine the optimal stopping point.

7.3. Combined Augmentation and Regularization Technique

7.3.1. Techniques and Motivations

Based on the findings from our individual augmentation and regularization experiments, we observed that Cutout and Gaussian Noise consistently delivered the strongest performance across datasets and noise levels. Cutout improves the network's ability to infer missing information by occluding regions of the input, which promotes contextual reconstruction. Gaussian Noise serves as a form of "noise inoculation," helping the model generalize to various noise patterns. Simultaneously, Early Stopping was shown to be the most effective regularization method in this context, halting training before the model begins to overfit to corrupted inputs. These techniques each enhance denoising performance through distinct yet complementary mechanisms - spatial masking, noise exposure, and adaptive training control.

To test whether their combination provides additive or synergistic benefits, we designed a joint ablation experiment applying both Cutout and Gaussian Noise simultaneously, with Early Stopping as the regularization strategy. This configuration was evaluated across all four datasets-MNIST, CIFAR-10, CIFAR-100, and STL-10-under all three Gaussian noise levels (15%, 25%, 50%). Results from this combined setting will be analyzed against the strongest-performing individual experiments to assess whether combining effective methods amplifies robustness and reconstruction accuracy in low-level denoising tasks.

7.3.2. Performance Metrics

Dataset	Noise %	Test Acc	Test Loss	Time (s)
MNIST	15%	0.961	0.0014	578
	25%	0.941	0.0029	570
	50%	0.900	0.0085	592
CIFAR-10	15%	0.953	0.0023	635
	25%	0.884	0.0042	775
	50%	0.746	0.0091	616
CIFAR-100	15%	0.952	0.0023	618
	25%	0.882	0.0042	640
	50%	0.746	0.0092	1164
STL-10	15%	0.952	0.0022	437
	25%	0.903	0.0037	447
	50%	0.802	0.0075	464

Table 17. Base Experiment Results Without Augmentation or Regularization

Dataset	Noise %	Test Acc	Test Loss	Time (s)
MNIST	15%	0.957	0.0017	544
	25%	0.939	0.0031	867
	50%	0.897	0.0088	873
CIFAR-10	15%	0.938	0.0027	687
	25%	0.876	0.0044	709
	50%	0.737	0.0094	816
CIFAR-100	15%	0.943	0.0025	670
	25%	0.878	0.0044	555
	50%	0.744	0.0093	676
STL-10	15%	0.927	0.0031	446
	25%	0.891	0.0041	446
	50%	0.789	0.0081	445

Table 18. Augmentation and Regularization Results Across Different Datasets and Noise Levels

7.3.3. Conclusions on Findings

At 15% noise, performance between the base and augmented models is similar, with the base model slightly ahead in most cases—for example, MNIST achieves 0.961 accuracy compared to 0.957 with augmentation. At 25% noise, augmentation offers minor improvements in datasets like CIFAR-100 and STL-10, while performance remains comparable for CIFAR-10 and MNIST. Under 50% noise, the augmented models consistently outperform the base, demonstrating improved robustness in high-noise environments.

Test losses also support the accuracy trends. Under lower noise, both approaches yield similar results, but as noise increases, the augmented models tend to produce lower test losses. This suggests better generalization capabilities when augmentation and regularization are applied.

Training time for augmented models is generally higher but

remains within a reasonable range. For instance, MNIST training time at 15% noise decreases slightly with augmentation (578s to 544s), but at 50% noise it increases to 873s. Despite the added time, the performance improvements justify the additional computational cost in high-noise scenarios.

7.4. Model Architecture

Table 19. Image Denoising Network Architecture — H = height of input image, W = width of input image, C_{in} = number of input channels, C_{out} = number of output channels.

Section	Layer	Configuration	Input Shape	Output Shape
Input				
	Input	—	—	$H \times W \times C_{in}$
Encoder				
	Conv2d + ReLU	$C_{in} \to 64$, 3×3, pad=1	$H \times W \times C_{in}$	$H \times W \times 64$
	Conv2d + ReLU	$64 \to 128$, 3×3, pad=1	$H \times W \times 64$	$H \times W \times 128$
	MaxPool2d	2×2, stride=2	$H \times W \times 128$	$\frac{H}{2} \times \frac{W}{2} \times 128$
Middle				
	Conv2d + ReLU	$128 \to 256$, 3×3, pad=1	$\frac{H}{2} \times \frac{W}{2} \times 128$	$\frac{H}{2} \times \frac{W}{2} \times 256$
	Conv2d + ReLU	$256 \to 256$, 3×3, pad=1	$\frac{H}{2} \times \frac{W}{2} \times 256$	$\frac{H}{2} \times \frac{W}{2} \times 256$
Decoder				
	ConvTranspose2d + ReLU	$256 \to 128$, 2×2, stride=2	$\frac{H}{2} \times \frac{W}{2} \times 256$	$H \times W \times 128$
	Conv2d + ReLU	$128 \to 64$, 3×3, pad=1	$H \times W \times 128$	$H \times W \times 64$
	Conv2d	$64 \to C_{out}$, 3×3, pad=1	$H \times W \times 64$	$H \times W \times C_{out}$
Output				
	Output	—	$H \times W \times C_{out}$	$H \times W \times C_{out}$

Dataset Information:

Dataset	Image Size	Channels	Classes
MNIST	28×28	1 (grayscale)	10 digits (0-9)
CIFAR10	32×32	3 (RGB)	10 classes
CIFAR100	32×32	3 (RGB)	100 classes
STL10	96×96	3 (RGB)	10 classes

7.5. Source Code

The full source code for our project is available on GitHub.