Robustness Analysis of Deep Learning Models Under Controlled Image Corruptions

Experimental Study

Project Repository: `https://github.com/soufianboukir/robust-vision-benchmark`

Dashboard URL: `https://robust-vision-benchmark.streamlit.app`

Soufian Boukir

Abstract

This study evaluates how robust different deep learning models (MLPs, CNNs, and ResNets) are when images are corrupted. We tested these models on CIFAR-10 using various real-world corruptions (noise, blur, compression, etc.) at different severity levels. We measured performance using standard metrics (accuracy, precision, recall, F1) plus robustness-specific ones (accuracy Drop, and sensitivity). Key finding: While deeper models achieved better accuracy on clean images, they were surprisingly less stable and durable when faced with corrupted inputs compared to shallow architectures.

Keywords: Robustness, image corruption, CIFAR-10, deep learning, CNN

1 Introduction

Machine learning and deep learning models show great performance across many computer vision tasks. However, models trained on curated image datasets can show different performance when the distribution of real-world images changes from that of the training data.

In practice, images are regularly affected by blur, noise, occlusion, compression artefacts, and lighting variation, all of these can influence model performance even when those models achieve great results on clean data.

In this work, the study explores the relative robustness of several deep learning-based image classification models under controlled image corruptions. The evaluation focuses on Multi-Layer Perceptrons (MLP-3 and MLP-5), Convolutional Neural Networks (LeNet-5 and AlexNet), and a Residual Network (ResNet-18), all tested on the CIFAR-10 dataset. These models are assessed under multiple artificial corruption types and severity levels, simulating realistic image degradation ranging from mild distortions to severe perturbations. The objective is to analyze how architectural differences influence both classification performance and robustness under distribution shifts.

2 Background and Related Work

2.1 Image Classification Models

Image classification is now used in many areas, such as facial recognition, city planning, etc. At first, traditional machine learning algorithms like logistic regression, KNN, and Random Forest were the best at this task, but they had trouble with data that had a lot of dimensions. Multilayer Perceptrons (MLPs) are feedforward networks with hidden layers that allow for non-linear learning. However, they turn images into one-dimensional vectors, which means they lose their spatial structure. Because of this, they are mostly used as baseline models instead of cutting-edge solutions (Goodfellow et al., 2016).

Because of this limit, we switched to Convolutional Neural Networks (CNNs), which are made just for images. Convolutional filters in CNNs take out local spatial

features like edges and textures. This makes them more efficient and better at their jobs by using spatial locality and parameter sharing (Yann LeCun, 1998).

As networks became deeper, training became increasingly difficult. Residual Networks (ResNets) addressed this by having layers learn residual functions rather than direct mappings, making optimization easier and enabling effective training of very deep architectures. ResNets demonstrated exceptional performance on large-scale tasks like ImageNet, showing that greater depth significantly improves classification accuracy without proportional computational increases (Kaiming He et al., 2015).

2.2 Robustness in Image Classification

Robustness in image classification refers to a model's ability to maintain reliable predictions when inputs are modified with noise, blur, or compression. This is critical when deploying models in real-world scenarios. However, deep neural networks often suffer significant performance drops when exposed to adversarial inputs, common corruptions, or distribution shifts—a major challenge in deep learning research. Current evaluation methods are limited due to rapid development, diverse corruption types, and simplistic metrics (Chang Liu et al., 2023).

3 Methodology

3.1 Dataset (CIFAR-10)

The CIFAR-10 dataset is a widely used benchmark dataset in computer vision research, containing a total of 60,000 color images with a resolution of 32x32 pixels distributed across 10 distinct object classes.

Each class contains exactly 6,000 images, ensuring a balanced representation across all categories. The dataset is partitioned into two primary subsets: a training set consisting of 50,000 images and a test set containing 10,000 images.

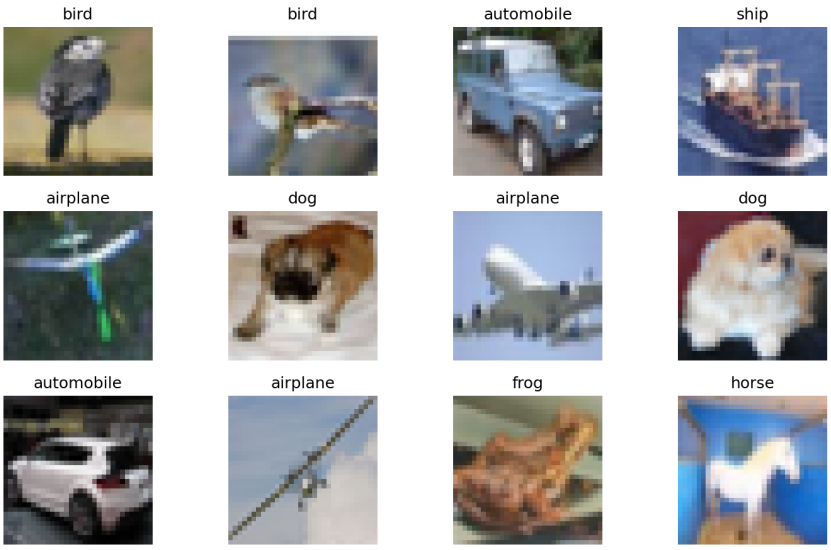

Figure 1: Sample overview of the CIFAR-10 dataset.

3.2 Corruption Engine

To evaluate robustness, a corruption pipeline is applied to the test data. Each corruption is applied at multiple severity levels, where higher severity corresponds to stronger degradation of the input image.

Six types of corruptions are considered:

3.2.1 Gaussian noise

Gaussian noise in images is a statistical noise model in which random values drawn from a Gaussian distribution are added to pixel intensities.

For an original image $I(x, y)$, the noisy image $I'(x, y)$ is modeled as:

$$I'(x, y) = I(x, y) + n(x, y) \ / \ n(x, y) \sim \mathcal{N}(0, \sigma^2)$$

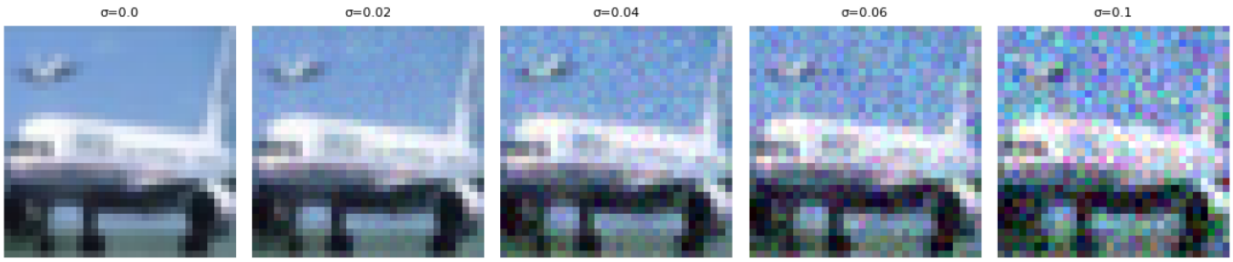

Figure 2: Visualization of gaussian noise different severities.

3.2.2 Gaussian blur

Gaussian blur smooths an image by replacing each pixel with a weighted average of its neighbors using a Gaussian kernel, where central pixels have higher influence.

The Gaussian kernel is defined as:

$$G(x, y) = \frac{1}{2\pi\sigma^2} \exp\left(-\frac{x^2 + y^2}{2\sigma^2}\right)$$

where σ controls the amount of smoothing.

In practice, a discrete and normalized kernel $K(i, j)$ is constructed and applied to the image using convolution:

$$I'(x, y) = \sum_i \sum_j I(x - i, y - j)\, K(i, j)$$

For RGB images, the convolution is applied independently to each channel.

Increasing σ produces stronger blur, progressively removing high-frequency details and degrading the visual structure of the image.

Figure 3: Visualization of gaussian blur different severities.

3.2.3 Random Occlusion

Occlusion is a structural image corruption in which a contiguous region of the image is covered or removed.

Let the original image be:

$$I(x, y) \in \mathbb{R}^{H \times W \times C}$$

The corrupted image $I'(x, y)$ is defined as:

$$I'(x, y) = \begin{cases} 0, & (x, y) \in \Omega, \\ I(x, y), & \text{otherwise} \end{cases}$$

where:

- $\Omega \subset \mathbb{Z}^2$ is a randomly selected rectangular region
- the size of Ω is controlled by the corruption severity

Figure 4: Visualization of random occlusion different severities.

3.2.4 JPEG Compression

JPEG compression is a lossy image corruption process that reduces image quality by removing visual information that is less perceptible to the human eye. In the context of robustness analysis, it simulates real-world degradations caused by image storage, transmission, or resizing on low-bandwidth systems.

Figure 5: Visualization of JPEG compression different severities.

3.2.5 Brightness Adjustment

Brightness adjustment is an intensity transformation that uniformly shifts pixel values across the entire image.

Let the image be:

$$I(x, y, c)$$

where:

- x, y are spatial coordinates

- c is the channel index for the RGB image

The brightness-adjusted image $I'(x, y, c)$ is defined as:

$$I'(x, y, c) = I(x, y, c) + \delta$$

where:

- $\delta \in \mathbb{R}$ is the brightness shift

Severity-controlled parameter

In this implementation:

$$\delta \in \{0.0, 0.07, 0.14, 0.21, 0.28\}$$

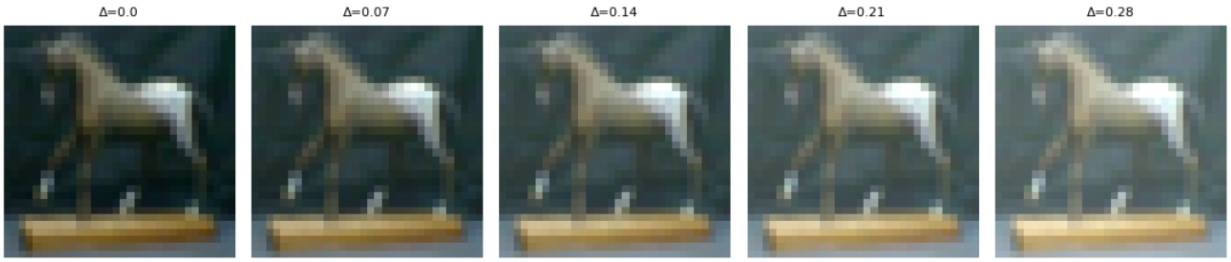

Figure 6: Visualization of brightness adjustment different severities.

3.2.6 Contrast Adjustment

Contrast adjustment is an intensity transformation that modifies the difference between pixel values by scaling their deviation from a reference intensity level.

The contrast-adjusted image $I'(x, y, c)$ is defined as:

$$I'(x, y, c) = (I(x, y, c) - m)\alpha + m$$

where:

- $\alpha \in \mathbb{R}$ is the contrast scaling factor
- $m = 0.5$ is the reference intensity (gray level)

Severity-controlled parameter

In this implementation:

$$\alpha \in \{1.0, 0.85, 0.75, 0.65, 0.55\}$$

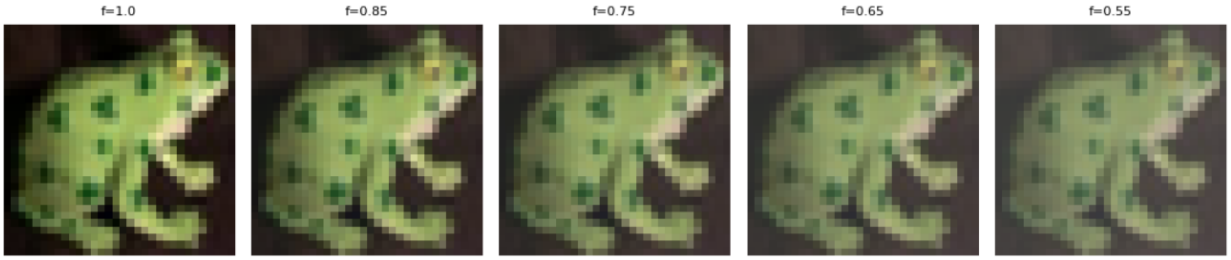

Figure 7: Visualization of contrast adjustment different severities.

3.2.7 Rotation Corruption

Rotation is a geometric corruption that modifies the spatial arrangement of pixels by rotating the image around its center.

Let the original image be:

$$I(x, y) \in \mathbb{R}^{H \times W \times C}$$

The rotation transformation is defined as:

$$\begin{pmatrix} x' \\ y' \end{pmatrix} = R(\theta) \begin{pmatrix} x - c_x \\ y - c_y \end{pmatrix} + \begin{pmatrix} c_x \\ c_y \end{pmatrix}$$

where:

$$R(\theta) = \begin{pmatrix} \cos(\theta) & -\sin(\theta) \\ \sin(\theta) & \cos(\theta) \end{pmatrix}$$

- $(c_x, c_y) = \left(\frac{W}{2}, \frac{H}{2}\right)$ is the image center,
- θ is the rotation angle in degrees,
- (x', y') are the rotated coordinates.

The corrupted image $I'(x, y)$ is obtained by interpolating pixel values at the rotated coordinates using bilinear interpolation.

Severity-controlled parameter: In this implementation:

$$\theta \in \{0°, 10°, 20°, 30°, 45°\}$$

Rotation corruptions at higher angles introduce black padding regions and significantly disrupt spatial structure, testing model invariance to geometric transformations.

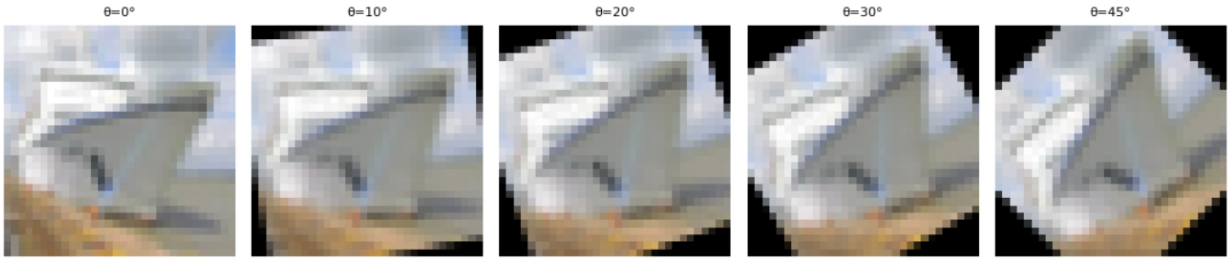

Figure 8: Visualization of rotation corruption at different severity levels. Rotation angles range from 0° (no rotation) to 45° (severe rotation). Note the progressive introduction of black padding regions as rotation angle increases, which disrupts spatial structure and challenges model invariance to geometric transformations.

3.3 Model Architectures

Table 1: Comparison of model architectures used in the experiments.

Model	Type	Depth	Preprocessing	Notes
MLP-3	MLP	3 layers	Normalization + Dropout	Fully connected baseline without spatial structure
MLP-5	MLP	5 layers	Normalization + Dropout	Deeper MLP with higher capacity
LeNet-5	CNN	2 conv + 3 FC	Normalization	Early CNN with simple feature extraction
AlexNet	CNN	5 conv + 3 FC	Normalization + Dropout	Deeper CNN with improved feature learning
ResNet-18	CNN	18 layers	Normalization	Residual network with skip connections

3.4 Training Configuration and Hyperparameters

Tuning hyperparameters is an important part of training deep learning models because these parameters have a direct effect on how quickly the model converges, how well it generalizes, and how efficiently it uses computing resources. When choosing hyperparameters for the CIFAR-10 dataset, we use a mix of best practices from the literature, the limitations of the Google Colab environment, and the unique features of each architecture.

Table 2: Training configuration and hyperparameters for DL models.

Model Architecture	Base Learning Rate	Batch Size	Epochs	Optimizer
MLP-3	1×10^{-3}	128	30	Adam
MLP-5	1×10^{-3}	128	30	Adam
LeNet-5	1×10^{-3}	128	50	Adam
AlexNet	1×10^{-3}	128	50	Adam
ResNet-18	1×10^{-3}	128	50	Adam

The training configuration is kept consistent across models to ensure a fair comparison of architectures under identical optimization conditions. All models are trained using the Adam optimizer with a fixed learning rate of 1×10^{-3} and a batch size of 128. The number of training epochs varies slightly depending on model complexity, with simpler architectures such as MLPs requiring fewer epochs, while deeper convolutional models and ResNet-18 are trained for longer to ensure convergence. This setup allows performance differences to be attributed primarily to architectural design rather than differences in training strategy.

4 Experimental Results and Robustness Analysis

This section presents comprehensive experimental results obtained from evaluating those models on the CIFAR-10 dataset. The evaluation encompasses two complementary assessment frameworks: clean performance evaluation (performance on the original, unmodified test set) and robustness evaluation (performance under controlled distribution shifts via image corruptions).

The analysis is structured around the following evaluation questions:

- Which models achieve the highest baseline accuracy on unmodified data?
- Which type of corruption degrades model performance the most?
- Are deeper models more robust than shallow ones?

4.1 Clean Performance Analysis

Clean performance evaluation establishes the baseline capability of each model under ideal conditions—when input data is unmodified and matches the training distribution.

Table 3: Clean performance ranking of all evaluated models on the CIFAR-10 test set.

Rank	Model	Accuracy	Precision	Recall	F1	Training time(m)
1	**ResNet-18**	0.8591	0.8590	0.8591	0.8589	11.40
2	**AlexNet**	0.7986	0.7986	0.7986	0.7985	8.44
3	**leNet-5**	0.5921	0.5889	0.5921	0.59	7.42
4	**MLP-5**	0.5543	0.5537	0.5543	0.5528	5.1
5	**MLP-3**	0.5411	0.5374	0.5411	0.5381	3.7

Note: MLP-3 and MLP-5 were trained on a Google Colab T4 GPU, while all other models were trained on an NVIDIA A100 GPU. Training times for MLP models on A100 are estimated based on hardware scaling.

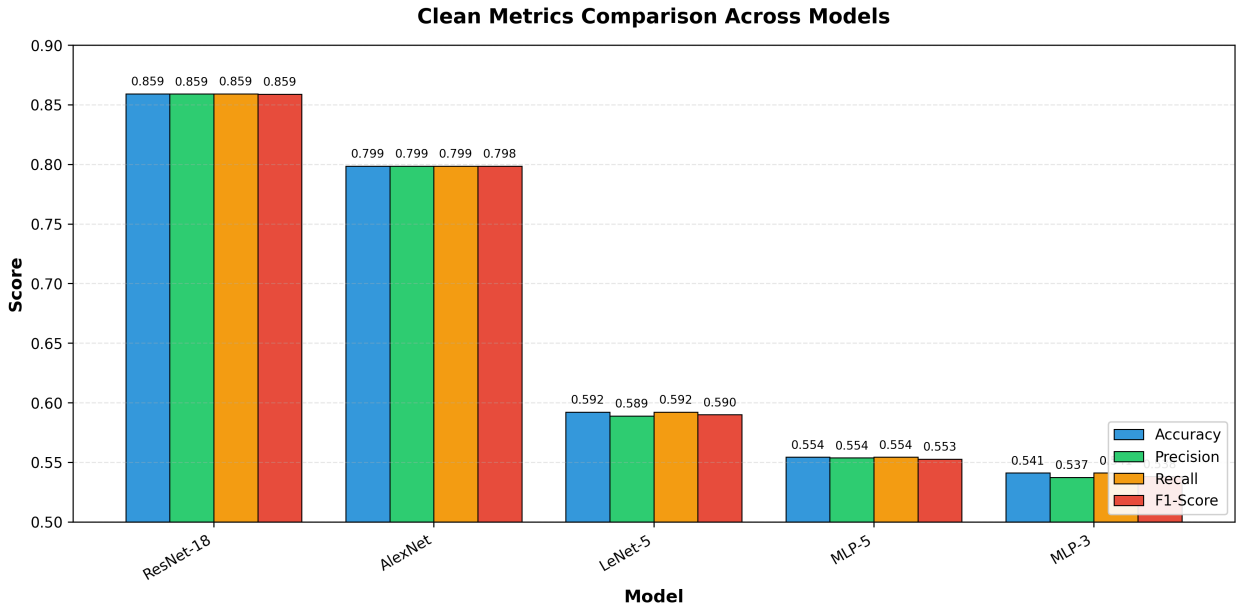

Figure 9: Clean Metrics Comparison Across Models.

Based on previous results (table, bar chart), we compared four evaluation metrics: accuracy, precision, recall, and F1-score across five models: ResNet-18, AlexNet, LeNet-5, MLP-5, and MLP-3 on the CIFAR-10 test set under clean conditions.

The chart shows that the ranking stays the same across all metrics. ResNet-18 performs the best, with all four metrics coming together at about 0.859. This shows that it can make accurate predictions and perform well across all classes. AlexNet follows as the second-best model, maintaining stable scores near 0.798 across all metrics, suggesting reliable but less expressive feature extraction compared to ResNet-18.

LeNet-5's performance drops noticeably, with metrics around 0.59. This is because it doesn't have the architectural capacity to handle CIFAR-10 complexity; it was made for simpler datasets like MNIST and doesn't have the depth needed for that.

The MLP-5 and MLP-3 models based on MLP are the worst of all the models tested, with scores between 0.54 and 0.55. Their lower performance is consistent across all metrics, which shows that fully connected architectures that don't use spatial feature extraction aren't good for image classification tasks.

4.2 Impact of Input Corruptions on Model Robustness

To evaluate model robustness beyond clean performance, we analyze how different input corruptions affect predictive accuracy across all models. While clean accuracy reflects a model's performance under ideal conditions, real-world deployment often involves distorted or noisy inputs. Therefore, this section focuses on quantifying performance degradation under various corruption types. The objective is to identify which corruptions have the most significant impact on model behavior and to compare sensitivity patterns across different architectures.

Metrics Used

Accuracy Drop (per corruption)

Accuracy drop measures the absolute degradation in performance relative to clean accuracy:

$$\Delta_{\text{acc}}^{(c)} = A_{\text{clean}} - A_{\text{corrupt}}^{(c)}$$

where:

- A_{clean} is the accuracy on clean data,
- $A_{\text{corrupt}}^{(c)}$ is the worst accuracy under corruption type c,
- $\Delta_{\text{acc}}^{(c)}$ is the accuracy drop caused by corruption c.

Higher values of $\Delta_{\text{acc}}^{(c)}$ indicate stronger performance degradation.

Sensitivity Score (normalized degradation)

Sensitivity score expresses the relative performance drop normalized by clean accuracy:

$$S^{(c)} = \frac{A_{\text{clean}} - A_{\text{corrupt}}^{(c)}}{A_{\text{clean}}}$$

where:

- $S^{(c)}$ is the sensitivity score for corruption type c.

- values closer to 1 indicate a highly damaging corruption,
- values closer to 0 indicate stronger robustness to that corruption.

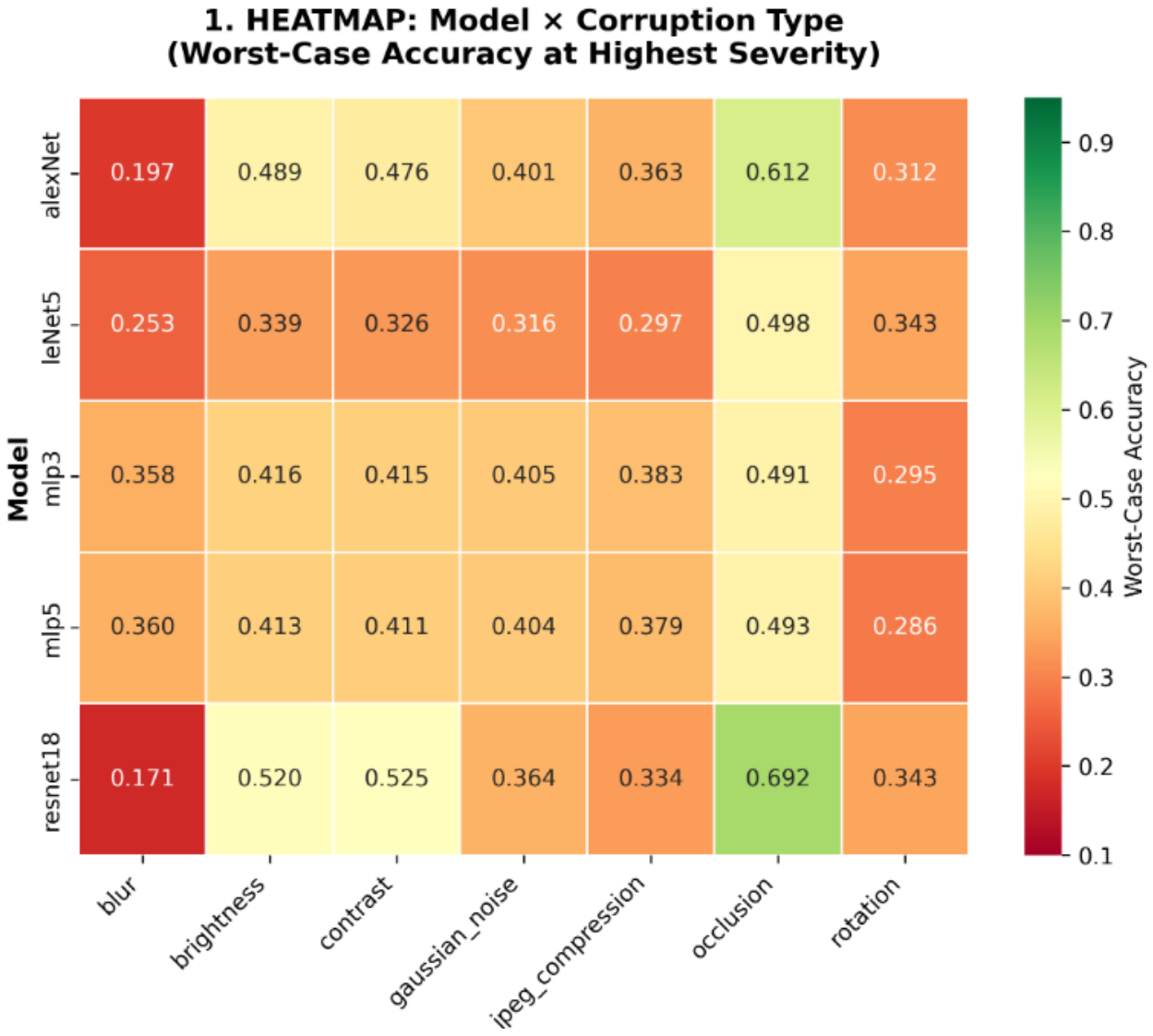

Figure 10: Worst case accuracy at heighest severity.

In Figure 10 we observe that blur is the most damaging corruption type across all models, with ResNet-18 achieving the lowest worst-case accuracy of 0.17. While

MLP-5 shows relatively better performance under blur (0.36 accuracy) compared to ResNet-18, it still experiences a 35.07(%) absolute accuracy drop from clean performance (Table 4), indicating significant degradation. Occlusion is the least harmful. Specific models excel with certain corruptions: ResNet18 performs well against contrast and occlusion issues, MLP5 resists rotation, and LeNet5 generally underperforms.

Table 4: Accuracy drop (%) from clean performance to worst-case corruption performance for each model.

Model	**Blur**	**Brightness**	**Contrast**	**Gaussian Noise**	**JPEG Compression**	**Occlusion**	**Rotation**
AlexNet	75.31	38.77	40.45	49.82	54.56	23.43	60.98
LeNet-5	57.25	42.83	44.89	46.72	49.87	15.94	42.02
MLP-3	33.76	23.08	23.25	25.17	29.26	9.28	45.44
MLP-5	35.07	25.49	25.78	27.21	31.68	10.97	48.40
ResNet-18	80.05	39.41	38.88	57.64	61.13	19.40	60.06

Table 5: Corruption impact analysis across all evaluated models.

Corruption	**Avg Worst Accuracy**	**Worst Model**	**Best Model**	**Avg Drop (%)**	**Sensitivity**
Blur	0.2680	ResNet-18	MLP-5	56.2891	0.3443
JPEG compression	0.3510	LeNet-5	MLP-3	45.3000	0.2955
Gaussian noise	0.3777	LeNet-5	MLP-3	41.3116	0.2727
Contrast	0.4307	LeNet-5	ResNet-18	34.6488	0.2139
Rotation	0.3158	MLP-5	LeNet-5	51.3825	0.2041
Brightness	0.4354	LeNet-5	ResNet-18	33.9172	0.1963
Occlusion	0.5572	MLP-3	ResNet-18	15.8044	0.0409

The results indicate that blur is the most destructive corruption across all evaluated models, exhibiting the highest sensitivity score and the largest average accuracy drop. This behavior can be attributed to the fact that blur removes high-frequency details and edge information, which are critical for convolutional feature extraction. In contrast, occlusion has the least impact, suggesting that models can still rely on partial object information for classification. JPEG compression and Gaussian noise introduce moderate degradation, affecting texture and pixel-level information without completely destroying spatial structure.

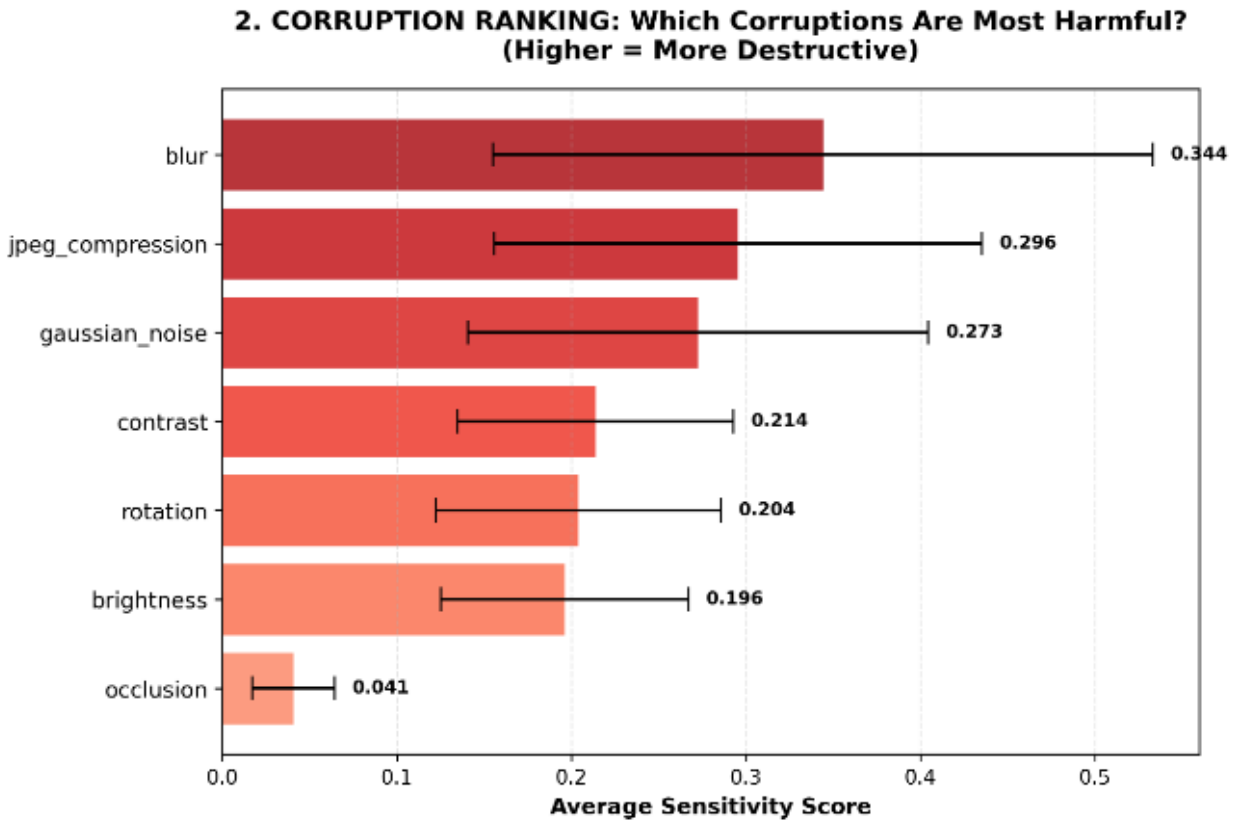

Figure 11: Worst-case accuracy of each model under different corruption types at the highest severity level. Lower values indicate stronger degradation.

Overall, the analysis demonstrates that structural corruptions such as blur have a significantly greater impact on model performance than intensity-based perturbations, highlighting a key limitation of current architectures in handling spatial degradation.

4.3 Overall Model Robustness Comparison

To complement the corruption-specific analysis, we now evaluate the overall robustness of each model across all corruption types. this analysis focuses on determining which models are more resilient under adverse conditions. To achieve this, we aggregate performance across all corruptions using worst-case accuracy and average performance degradation metrics, providing a global view of model robustness.

Metric Used: Average Worst-case Accuracy

$$A_{\text{worst}}^{(m)} = \frac{1}{C} \sum_{c=1}^{C} \min_{k} A^{(m,c,k)}$$

where:

- m denotes the model,
- c denotes the corruption type,
- k denotes severity level,
- C is the number of corruption types.

Table 6: Overall model robustness ranking across all corruption types.

Model	Clean Accuracy	Avg Worst Accuracy	Avg Drop (%)	Min Accuracy	Max Accuracy
ResNet-18	0.8591	0.4215	50.9404	0.1714	0.6924
AlexNet	0.7986	0.4069	49.0448	0.1972	0.6115
MLP-3	0.5411	0.3948	27.0349	0.2952	0.4909
MLP-5	0.5543	0.3923	29.2286	0.2860	0.4935
LeNet-5	0.5921	0.3387	42.7896	0.2531	0.4977

In Table 6 we observe that ResNet-18 has the best clean accuracy but the worst performance drop when there is corruption, with an average drop of more than 50(%). Even so, it still has competitive worst-case accuracy, which means it has strong feature representation but is very sensitive to severe distortions. AlexNet shows a similar pattern, with slightly worse clean performance but similar robustness traits.

MLP-based models, on the other hand, show much smaller drops in performance, which means they are more stable when the data is corrupted. But their lower clean accuracy makes them less useful overall. This shows that there is a trade-off between accuracy and robustness. Simpler models are more stable, but they don't have as much expressive power.

LeNet-5 does okay in both clean and corrupted settings, but it doesn't do great in either, which means it can't handle a lot of different visual variations.

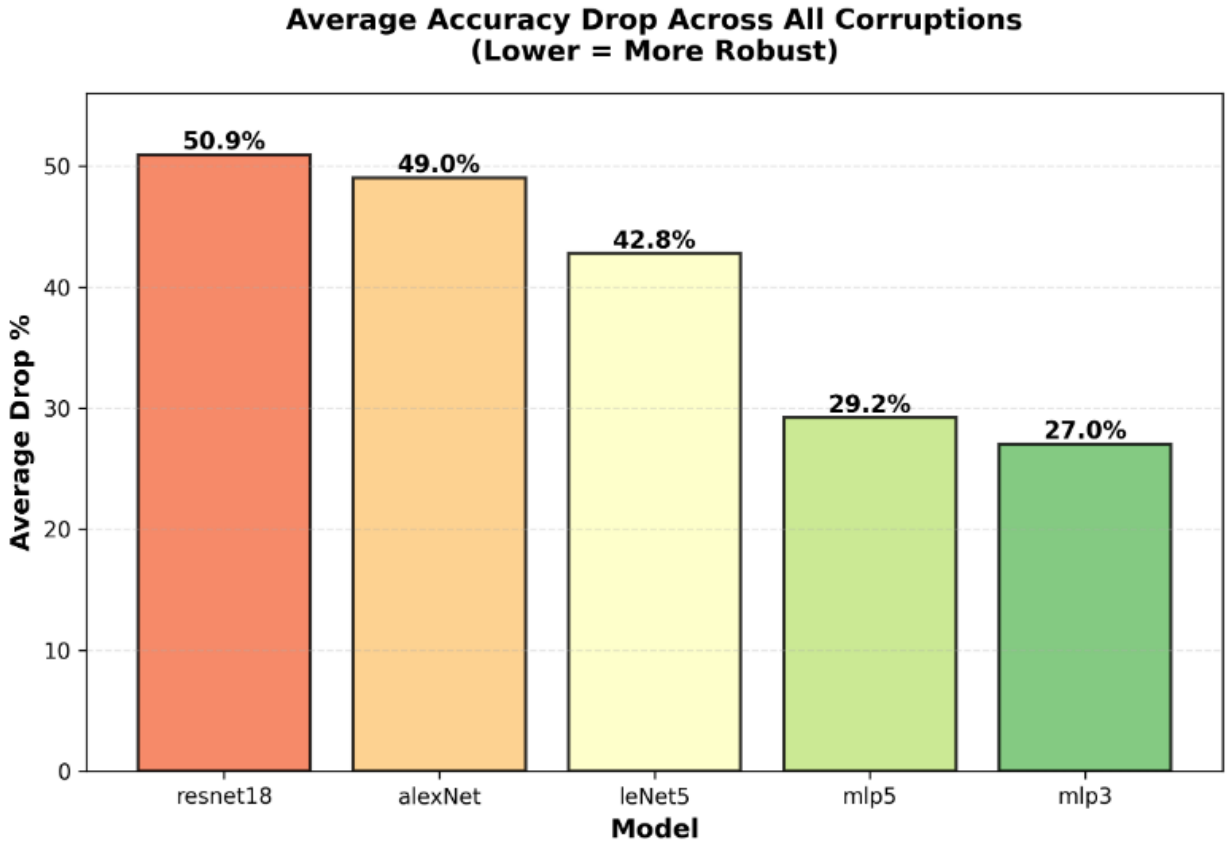

Figure 12: Comparison of model robustness based on average accuracy drop across all corruption types. Lower values indicate more robust models.

One important thing to note from this analysis is that having a higher clean accuracy doesn't always mean that the model is more robust. Deep convolutional models like ResNet-18 work better when everything is perfect, but they are more likely to break down when the structure is changed, like when it gets blurry. On the other hand, simpler architectures like MLPs show more stable degradation patterns, but they are less accurate overall. This trade-off shows a big problem with current deep learning models: better accuracy doesn't always mean better robustness when the distribution changes.

Overall, ResNet-18 remains the strongest model in terms of absolute performance, while MLP-based models demonstrate relatively higher robustness stability, emphasizing the trade-off between accuracy and resilience to input perturbations.

5 Discussion and Conclusion

This study examined the performance and robustness of various deep learning models on the CIFAR-10 dataset in both clean and corrupted images. While initial evaluation on clean data provides a baseline for model accuracy; it does not reflect real-world scenarios where input data is often degraded. Therefore, this study extended the analysis to evaluate model behavior under various corruption types, enabling a more comprehensive assessment of model reliability.

The results on clean data show a clear hierarchy in model performance, with deep convolutional architectures doing better than simpler ones. ResNet-18 had the highest accuracy, followed by AlexNet. MLP-based models and LeNet-5, on the other hand, did much worse. This confirms that architectures that can use spatial structure and hierarchical feature extraction work better for classifying images.

However, when the model performances were tested in a corrupt environment, there was a noticeable drop in the performance for all the models. we observed during the analysis that the level at which corruptions affect models is not uniform. The structural corruptions, especially the blurring, were found to have the highest negative impact on the performance of the models, this can be attributed to the loss of high-frequency details and edge information, which are critical for convolutional feature extraction. Occlusions, contrast caused the lowest negative effect on the models,suggesting that models can still rely on partial object information for classification.

A key finding of this study is the trade-off between clean accuracy and robustness. Although the Resnet-18 model was the most accurate under perfect conditions, it also showed the greatest sensitivity towards harsh corruptions. On the other hand, less complex models like MLP-3 and MLP-5 were more robust to the corruptions but suffered reduced performance relative to their baseline levels. This indicates that higher model capacity and complexity do not necessarily guarantee robustness under distribution shifts.

These results highlight fundamental differences in how model architectures process visual information. CNNs depend highly on spatial consistency and local features; hence, making them vulnerable to distortions that disrupt image structure. Fully connected models, which have low capacity for representation, are less sensitive to certain structural changes, leading to higher stability but lower accuracy.

Despite these findings, some limitations can be highlighted concerning the presented results. For instance, the analysis is based only on certain forms of corruption and degrees of distortion, which may not cover all types present in reality. Moreover, robustness was evaluated in terms of classification accuracy without taking into consideration such factors as uncertainty estimation or calibration. Future work could extend this analysis to include additional robustness metrics and more complex datasets.

In conclusion, this study demonstrates that while modern deep learning models achieve high accuracy on clean data, they remain vulnerable to input corruptions, particularly those affecting spatial structure. The results emphasize the importance of evaluating models beyond standard benchmarks and highlight the need for robustness-aware model design in real-world applications.

References

Goodfellow, I., Bengio, Y., & Courville, A. (2016). Deep learning. MIT Press.

He, K., Zhang, X., Ren, S., & Sun, J. (2015). Deep residual learning for image recognition. In 2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR) (pp. 770-778). IEEE. https://doi.org/10.1109/CVPR.2015.7298965

LeCun, Y. (1998). Gradient-based learning applied to document recognition. Proceedings of the IEEE, 86(11), 2278-2324. https://doi.org/10.1109/5.726791

Liu, C., Dong, Y., Xiang, W., Yang, X., Su, H., Zhu, J., Chen, Y., He, Y., Xue, H., & Zheng, S. (2023). A comprehensive study on robustness of image classification models: Benchmarking and rethinking. arXiv preprint arXiv:2302.14593.