# AI6103 Group Project: Rethinking Convolutional Network Design for Image Restoration Reimplementation and Improvements

**Yu Yongshan**
**Matriculation ID: G2302824H**
s230058@e.ntu.edu.sg
**Yip Chen Fei**
**Matriculation ID: G2304486K**
YIPC0010@e.ntu.edu.sg
**Wong Seik Man**
**Matriculation ID: G2303494L**
SWONG070@e.ntu.edu.sg
**Kee Ming Yuan**
**Matriculation ID: G2304842E**
MKEE004@e.ntu.edu.sg

[1]Nanyang Technological University
School of Computer Science and Engineering (SCSE)
50 Nanyang Avenue, Singapore 639798

## Contents

**Introduction** 1

**Model Foundations and Dataset Overview** 1
- Model Architecture . . . 1
- Loss Functions . . . 2
- Performance Metrics: PSNR and SSIM . . . 2
- Dataset Description . . . 2

**Reimplementation** 2

**Improvements** 3
- Weight Decay . . . 3
- Mixup . . . 4
- Nesterov Momentum Optimization . . . 5

**Conclusion** 6

**Abstract**

This report focuses on the reimplementation and enhancement of the IRNeXt model, originally presented in the paper "IRNeXt: Rethinking Convolutional Network Design for Image Restoration" written by Yuning Cui, Wenqi Ren, Sining Yang, Xiaochun Cao, and Alois Knoll in ICML 2023 Conference [1]. Our objective is to replicate the model's experiments and improve the model performance on the image dehaze restoration task, and propose improvements aimed at enhancing the key performance metrics of Peak Signal-to-Noise Ratio (PSNR) and Structural Similarity Index (SSIM).

## Introduction

Image restoration has made notable progress with the development of advanced computational models. Among these, the IRNeXt model, created by Yuning Cui et al., is recognized for its proficiency in handling diverse image restoration tasks. The focus of this report on reimplementing IRNeXt for image dehazing is primarily driven by the exclusive availability of the RESIDE dataset for dehazing in the authors' GitHub repository, while the access to other datasets is not provided.

Our reimplementation closely follows the original study's methodology, applying it to image dehazing, and assesses potential areas for refinement. By replicating the original experiments and exploring possible improvements, this report contributes to a deeper understanding of the IRNeXt model and its application in image restoration.

## Model Foundations and Dataset Overview

### Model Architecture

**3.1. Overall Architecture** The IRNeXt model utilizes a U-shaped architecture, beginning with a $3 \times 3$ convolution layer for processing image $\mathbf{I} \in \mathbb{R}^{3\times H\times W}$. It comprises three CNNBlocks with residual blocks and a Multi-Scale Module (MSM) for diverse scale learning (Figure 1(c)), supplemented by a ConvS module for low-resolution feature extraction (Figure 1(b)). The architecture concludes by merging the enhanced output with the original degraded image, ensuring detailed restoration.

**3.2. Multi-Scale Module** The MSM tackles various blurs using a multi-stage encoder-decoder network. It processes input $\mathbf{X} \in \mathbb{R}^{H\times W}$ with average pooling at different scales, refining each scale's output through the Local Attention Module (LAM) for filter modulation (Figure 1(d)). The scale-aligned outputs are formulated as:

$$\hat{\mathbf{X}}_i = \mathrm{LAM}\left(\mathrm{AP}_{2^{4-i}}(\mathbf{X}) + \hat{\mathbf{X}}_{i-1} \uparrow_2\right) \uparrow_{2^{4-i}}, \quad (1)$$

$$\hat{\mathbf{X}} = \mathrm{Conv}_{3\times 3}\left(\sum_{i=1}^{3} \hat{\mathbf{X}}_i + \mathbf{X}\right). \quad (2)$$

**3.3. Local Attention Module** LAM, integral to each MSM branch, refines multi-scale features. It generates dynamic attention weights with a convolution block, substituting Softmax with Tanh for efficient attention modulation (Figure 1(e)). The weights are applied group-wise for frequency-specific emphasis:

$$\mathbf{A} = \text{Tanh}\left(\text{Conv}_{1\times1}\left(\text{GAP}\left(\text{Conv}_{3\times3}(\mathbf{X})\right)\right)\right), \quad (3)$$

$$\hat{\mathbf{X}}_{g,h,w} = \sum_{i=0}^{K-1}\sum_{j=0}^{K-1} \mathbf{X}_{g,h-\lfloor\frac{K}{2}\rfloor+i,w-\lfloor\frac{K}{2}\rfloor+j}\mathbf{A}'_{g,i,j} + \mathbf{X}_{g,h,w}. \quad (4)$$

These features are vital for the model's advanced image restoration capabilities.

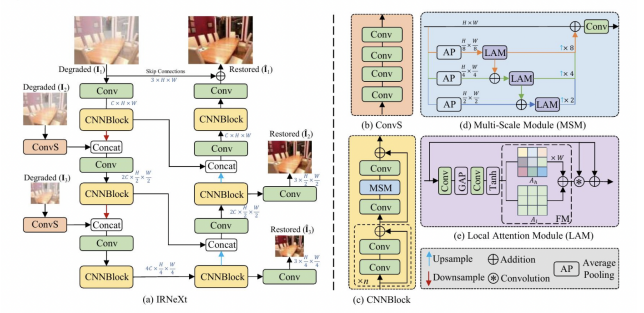


Figure 1: The architecture of IRNeXt: **(a)** depicts the overall U-shaped architecture with multi-input and multi-output design; **(b)** shows the ConvS module for feature extraction from low-resolution images; **(c)** details a CNNBlock containing multiple residual blocks with the MSM; **(d)** illustrates the MSM for multi-scale learning; and **(e)** displays the LAM for information aggregation.

## Loss Functions

The IRNeXt model employs a dual-domain loss function, combining spatial and frequency domains, crucial for integrating Filter Modulation in the Local Attention Module. The spatial loss (5) and frequency loss (6) are:

$$\mathcal{L}_{\text{spatial}} = \sum_{i=1}^{3} \frac{1}{P_i} \left\| \hat{\mathbf{I}}_i - \mathbf{Y}_i \right\|_1, \quad (5)$$

$$\mathcal{L}_{\text{frequency}} = \sum_{i=1}^{3} \frac{1}{P_i} \left\| F\left(\hat{\mathbf{I}}_i\right) - F\left(\mathbf{Y}_i\right) \right\|_1, \quad (6)$$

Here, $\hat{\mathbf{I}}_i$ and $\mathbf{Y}_i$ denote predicted and ground truth images, respectively. $P_i$ is the number of elements per image, and $F$ is the fast Fourier transform. The total loss (7) combines these components:

$$\mathcal{L}_{\text{total}} = \mathcal{L}_{\text{spatial}} + \lambda \mathcal{L}_{\text{frequency}}, \quad (7)$$

with $\lambda = 0.1$ to balance spatial and frequency domain training.

## Performance Metrics: PSNR and SSIM

The IRNeXt model's effectiveness in image restoration is gauged using two decibel-measured metrics: Peak Signal-to-Noise Ratio (PSNR) and Structural Similarity Index (SSIM). PSNR assesses reconstruction quality, calculated as:

$$\text{PSNR} = 10 \cdot \log_{10}\left(\frac{\text{MAX}_I^2}{\text{MSE}}\right), \quad (8)$$

where $\text{MAX}_I$ is the maximum pixel intensity, and MSE is the mean squared error. SSIM evaluates perceptual quality, focusing on luminance, contrast, and structure:

$$\text{SSIM}(x,y) = \frac{(2\mu_x\mu_y + C_1)(2\sigma_{xy} + C_2)}{(\mu_x^2 + \mu_y^2 + C_1)(\sigma_x^2 + \sigma_y^2 + C_2)}, \quad (9)$$

Higher PSNR and SSIM values indicate better image restoration, with PSNR quantifying error levels and SSIM reflecting perceptual similarity to the original image.

## Dataset Description

The synthetic RESIDE dataset, illustrated in Figure 2, is employed for IRNeXt's dehazing performance analysis. RESIDE's ITS contains 13,990 hazy counterparts of 1,399 sharp images; OTS offers 313,950 hazy variants from 8,970 originals. SOTS, comprising 500 indoor and outdoor hazy images each, serves as the evaluation ground for models trained on ITS (**ITS models**) and OTS (**OTS models**).

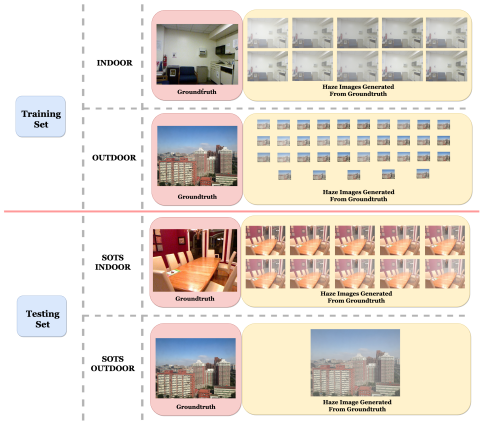


Figure 2: Example-based synthetic haze generation: The dataset is categorized into indoor and outdoor scenes for both training and testing sets. Despite identical ground truth images in the outdoor training and testing sets, the artificially induced haze differs in properties, reflecting varied atmospheric conditions.

## Reimplementation

**Experimental Setup** Our reimplementation of IRNeXt strictly followed the original protocol, using the authors' training and testing code on a Google Colab A100 GPU. The complete RESIDE dataset was utilized, training two model variants: one on the Indoor Training Set **(ITS)** and another on the Outdoor Training Set **(OTS)**, in alignment with the original experiment's methodology. The following parameters are used in our reimplementations:

Table 1: Reimplementation Parameters for ITS and OTS Models

| Parameter | ITS Model | OTS Model |
|---|---|---|
| Batch Size | 4 | 8 |
| Learning Rate | $1 \times 10^{-4}$ | $1 \times 10^{-4}$ |
| Weight Decay | 0 | 0 |
| Number of Epochs | 300 | 30 |
| Number of Workers | 8 | 8 |
| Optimizer | Adam | Adam |
| Avg PSNR Calculation Interval | Every 10 epochs | Every 10 epochs |

**Performance on SOTS Datasets** We conducted a comparison on the SOTS-indoor and SOTS-outdoor datasets among our replicated ITS and OTS models, the original authors' best weights, and the performance of these best weights when run in our experimental setup. The outcomes are represented in Table 2, where the performances of our models are hightlighted by yellow, highest-performing metrics are emphasized in **bold** for quick reference; and Figure 3.

Table 2: Performance on SOTS Datasets

| Model | PSNR | SSIM |
|---|---|---|
| **SOTS-Indoor Testing Set** | | |
| **Our Best ITS Model** | **42.12** | **0.99670** |
| Authors' Best ITS Model (Our Env) | 39.65 | 0.99507 |
| Authors' Best ITS Model (Paper) | 41.21 | 0.99600 |
| **SOTS-Outdoor Testing Set** | | |
| **Our Best OTS Model** | **39.27** | **0.99588** |
| Authors' Best OTS Model (Our Env) | 37.67 | 0.99458 |
| Authors' Best OTS Model (Paper) | 39.18 | 0.99600 |

**Discussions** In our analysis of IRNeXt models, we noted PSNR variations within expected statistical ranges and consistent SSIM values. Our models showed a PSNR increase of approximately +1, while using the original authors' weights resulted in a −2 PSNR decrease (see Table 2 and Figure 3).

To assess the normalcy of these variances, we conducted an 100-iteration bootstrap samplings on the ITS and OTS models' test logs, each containing 500 results. This involved repeatedly selecting and reselecting one result from the log 500 times per iteration. The resulting 95% confidence intervals for PSNR were $(39.267, 42.275)$ for ITS and $(37.104, 40.018)$ for OTS models. The inclusion of the authors' best models' PSNR in these intervals in our tests confirms that the PSNR changes are within expected limits, validating our replication's robustness and accuracy.

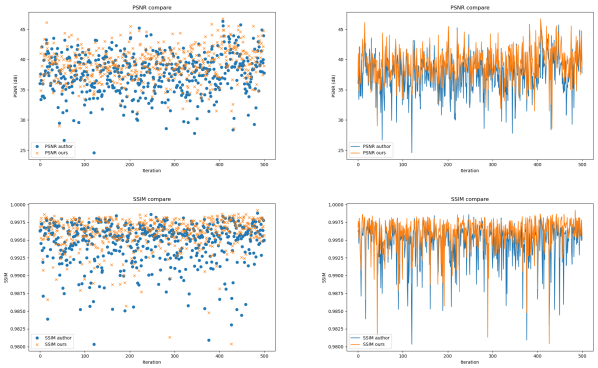


Figure 3: Comparison of PSNR and SSIM values over iterations between our replication (orange) and the original authors' weight in our environment (blue). **Left:** Dot plots. **Right:** Line plots.

## Improvements

Our report meticulously details each IRNeXt model iteration, closely following the original study's parameters, with deviations marked in light gray. Performance metrics, in yellow, contrast against the authors' results shown in the paper using ↑(increase), ↓(decrease), or ≈(similar performance) for performance changes, and highest metrics are **bolded**. Tests of authors' optimal model weights on non-corresponding datasets (e.g., ITS model on SOTS outdoor) are indicated in red, highlighting model adaptability.

In our experiments, performed on Google Colab with an A100 GPU, we replicate the original model's environment to precisely gauge the impact of our specific enhancements on image restoration. This approach, maintaining all settings except for our introduced improvements, allows for a controlled observation of performance changes, offering a clear insight into the efficacy of each optimization technique.

### Weight Decay

Weight decay (WD), as $L2$ regularization, is crucial for mitigating overfitting in enhancing the IRNeXt model [2, 3]. This technique modifies the loss function by adding a penalty term for large weights, formulated as:

$$\mathcal{L}_{\text{modified}} = \mathcal{L}_{\text{original}} + \frac{\lambda}{2} \sum_{w} w^2, \tag{10}$$

where $\mathcal{L}_{\text{original}}$ is the initial loss, $\lambda$ the weight decay coefficient, and $w$ the model weights [4]. The weight update rule incorporating $L2$ regularization is:

$$w_t = w_{t-1} - \eta \left( \nabla_w \mathcal{L}(w_{t-1}) + \lambda w_{t-1} \right), \tag{11}$$

where $\eta$ is the learning rate, and $\nabla_w \mathcal{L}(w_{t-1})$ the gradient of the loss with respect to weights [5]. Integrating weight decay encourages the model to favor

simpler, more generalizable solutions, reducing susceptibility to noise and anomalies [6]. This regularization is vital in complex architectures like IRNeXt, promoting smoother decision boundaries and enhancing the model's performance on new datasets, crucial for robust and generalizable image restoration [7].

**Experimental Setup** Despite training the IRNeXt model for 190 epochs (ITS) and 18 epochs (OTS) using Adam, performance metrics plateaued at PSNR $\leq 20$ and SSIM $\leq 0.58$ for both models, showing minimal improvement from the initial 30 (ITS) and 7 (OTS) epochs [8]. We hypothesize that Adam's integration with weight decay might lead to over-regularization, hindering improvement [9]. To address this and optimize resource use, we shifted to AdamW, expected to regulate weight decay and learning rates more effectively, crucial for image restoration [10].

Adam's approach of adapting learning rates per gradient moments likely amplifies regularization effects when merged with weight decay [11]:

$$w_t = w_{t-1} - \frac{\eta}{\sqrt{\hat{v}_t} + \epsilon} \hat{m}_t, \tag{12}$$

In contrast, AdamW decouples weight decay [9]:

$$w_t = w_{t-1} - \eta \left( \frac{\hat{m}_t}{\sqrt{\hat{v}_t} + \epsilon} + \lambda w_{t-1} \right), \tag{13}$$

where $\lambda$ directly impacts weights, suggesting more consistent regularization [12]. This switch to AdamW aims to overcome previous limitations and enhance model performance. AdamW's efficacy in the IRNeXt model is detailed in Table 3 and remains under evaluation [13].

Table 3: Training Parameters for ITS and OTS Models

| Parameter | ITS Model | OTS Model |
|---|---|---|
| Batch Size | 4 | 8 |
| Learning Rate | $1 \times 10^{-4}$ | $1 \times 10^{-4}$ |
| Weight Decay | $5 \times 10^{-4}$ | $5 \times 10^{-4}$ |
| Optimizer | AdamW | AdamW |
| Number of Epochs | 300 | 30 |
| Number of Workers | 8 | 8 |
| Avg PSNR Calculation Interval | Every 10 epochs | Every 10 epochs |

**Performance on SOTS Datasets** As shown in Table 4, PSNR and SSIM with Weight Decay and AdamW optimizer is higher than the author's indoor model in our environment but less than that stated in the paper for outdoor model. PSNR and SSIM are lower for outdoor model with Weight Decay and AdamW for both author's model in our environment and the paper.

**Discussion** In our experiments, ITS model testing mirrored the original authors', indicating effective learning and generalization (Table 4). However, the OTS model diverged significantly: excellent in training (PSNR $\geq 44.03$) but substantially lower in testing (PSNR 31.02), suggesting overfitting [14, 15].

Table 4: Performance on SOTS Datasets

| Model | PSNR | SSIM |
|---|---|---|
| **SOTS-Indoor Testing Set** | | |
| Our Best ITS Model with WD $5 \times 10^{-4}$ | 41.04 $\approx$ | 0.99576 $\approx$ |
| Authors' Best ITS Model (Our Env) | 39.65 | 0.99507 |
| **Authors' Best ITS Model (Paper)** | **41.21** | **0.99600** |
| **SOTS-Outdoor Testing Set** | | |
| Our Best OTS Model with WD $5 \times 10^{-4}$ | 31.02 $\downarrow$ | 0.98075 $\downarrow$ |
| Authors' Best OTS Model (Our Env) | 37.67 | 0.99458 |
| **Authors' Best OTS Model (Paper)** | **39.18** | **0.99600** |

This discrepancy likely arises from differences in dataset complexity and the impact of weight decay (WD $5 \times 10^{-4}$) combined with AdamW optimization. ITS, with simpler indoor scenes, benefited from WD and adaptive learning rates, enhancing generalization. In contrast, OTS, covering diverse outdoor scenes, might have been overly constrained by aggressive WD, leading to underfitting and performance drops [16, 6, 17].

Furthermore, AdamW's distinct approach to weight decay, decoupled from adaptive rates, might have varied effects based on dataset complexity. It potentially facilitated ITS's learning stability but restricted OTS's ability to capture complex dehazing features [9, 11].

## Mixup

Mixup is a data augmentation technique that blends two training images and their labels, creating mixed samples to enhance model generalization [18]. It generates new training examples $(\mathbf{x}', \mathbf{y}')$ by linearly interpolating pairs of original examples $(\mathbf{x}_i, \mathbf{y}_i)$ and $(\mathbf{x}_j, \mathbf{y}_j)$, formulated as:

$$\mathbf{x}' = \lambda \mathbf{x}_i + (1 - \lambda)\mathbf{x}_j, \quad \mathbf{y}' = \lambda \mathbf{y}_i + (1 - \lambda)\mathbf{y}_j, \tag{14}$$

where $\lambda$ is drawn from a Beta distribution, typically Beta$(\alpha, \alpha)$ [19].

Incorporating Mixup into the IRNeXt model, we aim to improve robustness and generalization, especially for image restoration with diverse degradation patterns [20]. Mixup's linear combinations of inputs and labels help the model learn more fluid decision boundaries, fostering adaptability to new data variations and reducing overfitting risks [21]. This approach, especially with a $\beta(0.2, 0.2)$ distribution, exposes the model to diverse training influences, enhancing its understanding of class relationships and feature space overlaps [22].

Mixup also aids in model calibration, encouraging more probabilistic predictions and mitigating overfitting, improving the model's resilience to input variations [23].

**Experimental Setup** We merged the Indoor Training Set (ITS) and Outdoor Training Set (OTS) into a single Combined Restoration Dataset **(CRD)** from the

RESIDE dataset, enriching it with diverse image degradation scenarios [24]. To enhance model exposure to varied interpolations and improve degradation pattern recognition, we applied mixup augmentation using a $\beta(0.2, 0.2)$ distribution, favoring samples strongly influenced by one mixed example for better generalization across image qualities [25]. Parameter adjustments, reflecting the CRD's size and diversity, were made based on the OTS model settings (details in Table 5).

Table 5: Training Parameters for CRD Model

| Parameter | CRD Model |
|---|---|
| Dataset | CRD |
| Batch Size | 8 |
| Learning Rate | $1 \times 10^{-4}$ |
| Mixup | $\beta(0.2, 0.2)$ |
| Optimizer | Adam |
| Number of Epochs | 30 |
| Number of Workers | 8 |
| Avg PSNR Calculation Interval | Every 10 epochs |

**Performance on SOTS Datasets** As seen in Table 6, PSNR and SSIM values trained from CRD dataset did not yield better results than the author's model when ran in our environment and that stated in the paper. We found that when testing in SOTS outdoor dataset using the author's best indoor model in our environment gives poorer results than our model. The same goes for testing author's best outdoor model in our environment with the SOTS indoor dataset.

Table 6: Performance on SOTS Datasets

| Model | PSNR | SSIM |
|---|---|---|
| **SOTS-Indoor Testing Set** | | |
| Our Best CRD Model with $\beta(0.2, 0.2)$ | 25.37 ↓ | 0.91050 ↓ |
| Authors' Best ITS Model (Our Env) | 39.65 | 0.99507 |
| **Authors' Best ITS Model (Paper)** | **41.21** | **0.99600** |
| Authors' Best OTS Model (Our Env) | 17.66 ↓ | 0.84197 ↓ |
| **SOTS-Outdoor Testing Set** | | |
| Our Best CRD Model with $\beta(0.2, 0.2)$ | 29.48 ↓ | 0.97455 ↓ |
| Authors' Best OTS Model (Our Env) | 37.67 | 0.99458 |
| **Authors' Best OTS Model (Paper)** | **39.18** | **0.99600** |
| Authors' Best ITS Model (Our Env) | 19.10 ↓ | 0.85695 ↓ |

**Discussion** Our experimentation with the CRD model, leveraging Mixup with a $\beta(0.2, 0.2)$ distribution, showcased its multitasking capabilities in image dehazing, particularly when trained on a dataset combining both indoor and outdoor images. This resulted in consistent performance across various SOTS datasets, highlighting the potential of Mixup in enhancing model adaptability across diverse environments. Although the CRD model did not outperform the authors' domain-specific models, its consistent performance in varied settings is indicative of its robust multitasking ability.

The disparity in performance of the original authors' ITS and OTS models in environments other than their specific training conditions, as reflected by the decreased PSNR and SSIM metrics, points towards a limitation in their multitasking capacity. This suggests that while specialization can be beneficial in some cases, it impedes performance in diverse, unaccustomed conditions.

In contrast, the balanced performance of the CRD model across various environmental conditions suggests a more versatile and adaptable approach. This robustness, not limited to specialized settings but extendable to varied conditions, demonstrates the effectiveness of the Mixup strategy in fostering a multitasking model.

Looking towards future potential, we envisage an approach where the CRD dataset, or even more diverse datasets encompassing a broader range of haze scenarios, could be further enhanced [26]. One promising direction could involve incorporating a specific feature within the dataloader that informs the model about the origins of the data [27, 28]. This would be a natural extension of the Mixup methodology [18], where the model is not only trained on a mixture of data from various sources but also made aware of these sources to better understand and adapt to the unique characteristics of each environment [19, 22]. Such an advancement would harness the strengths of Mixup and push the boundaries of model adaptability and generalization in the realm of image restoration [20, 21].

## Nesterov Momentum Optimization

Nesterov Momentum Optimization enhances standard momentum optimization by introducing a predictive update mechanism [29], as defined by the equations:

$$\mathbf{v}_t = \mu \mathbf{v}_{t-1} + \eta \nabla \mathcal{L}(\mathbf{w}_{t-1} - \mu \mathbf{v}_{t-1}), \tag{15}$$

$$\mathbf{w}_t = \mathbf{w}_{t-1} - \mathbf{v}_t. \tag{16}$$

where $\mathbf{w}_t$ and $\mathbf{v}_t$ are the parameter and velocity vectors at step $t$, $\mu$ is the momentum factor and $\eta$ the learning rate. Nesterov Momentum anticipates gradient calculation as shown in Equation 17:

$$w_{t+1} = w_t + \mu v_t - \alpha \nabla L(w_t + \mu v_t), \tag{17}$$

This approach computes the gradient $\nabla L$ at the anticipated position $(w_t + \mu v_t)$, enabling more nuanced parameter adjustments [30]. Such foresight is theorized to yield faster convergence and stabilize optimization in complex tasks like image restoration [31].

**Experimental Setup** In optimizing Nesterov Momentum value (M) for the IRNeXt model within the range 0-1, four distinct M values (0.2, 0.4, 0.6, 0.8) were evaluated over 20 epochs for ITS model, and 10 epochs

for OTS model. The criterion for selection was the highest average PSNR and SSIM on the validation set:

Table 7: PSNR and SSIM with different M Values

| **ITS Model** | | | | |
|---|---|---|---|---|
| **Momentum (M)** | 0.2 | 0.4 | 0.6 | 0.8 |
| **Average PSNR (dB)** | **12.24** | 11.13 | 12.06 | 10.26 |
| **Average SSIM (dB)** | **0.69106** | 0.67365 | 0.68955 | 0.64950 |
| **OTS Model** | | | | |
| **Momentum (M)** | 0.2 | 0.4 | 0.6 | 0.8 |
| **Average PSNR (dB)** | **16.42** | 14.91 | 16.40 | 16.14 |
| **Average SSIM (dB)** | **0.83260** | 0.81506 | 0.82642 | 0.82540 |

As shown in Table 7, for the ITS model, M=0.2 emerged as optimal, achieving an average PSNR of 12.24 dB and SSIM of 0.69106 dB. Similarly, the outdoor model showed the best performance with M=0.2, yielding an average PSNR of 16.12 dB and SSIM of 0.82260 dB. This suggests that a lower M value aligns better with the specific complexities and characteristics of both datasets, favoring gradual momentum changes.

The final training parameters incorporating M=0.2 for both datasets are detailed in Table 8.

Table 8: Training Parameters for ITS and OTS Models

| **Parameter** | **ITS Model** | **OTS Model** |
|---|---|---|
| Batch Size | 4 | 8 |
| Learning Rate | $1 \times 10^{-4}$ | $1 \times 10^{-4}$ |
| M | M=0.2 | M=0.2 |
| Optimizer | Adam | Adam |
| Number of Epochs | 300 | 30 |
| Number of Workers | 8 | 8 |
| Avg PSNR Calculation Interval | Every 10 epochs | Every 10 epochs |

**Performance on SOTS Datasets** As seen in Table 9, both PSNR and SSIM values are lower in indoor and outdoor model for both author's best model when ran in our environment and of that stated the paper.

Table 9: Performance on SOTS Datasets

| **Model** | **PSNR** | **SSIM** |
|---|---|---|
| **SOTS-Indoor (ITS Model)** | | |
| Our Best Model with M=0.2 | 12.61 ↓ | 0.69868 ↓ |
| Authors' Best ITS Model (Our Env) | 39.65 | 0.99507 |
| **Authors' Best ITS Model (Paper)** | **41.21** | **0.99600** |
| **SOTS-Outdoor (OTS Model)** | | |
| Our Best Model with M=0.2 | 16.24 ↓ | 0.83113 ↓ |
| Authors' Best OTS Model (Our Env) | 37.67 | 0.99458 |
| **Authors' Best OTS Model (Paper)** | **39.18** | **0.99600** |

**Discussion** Following our experiments with the IRNeXt model, we observed that prolonged training under Nesterov Momentum (M=0.2) did not significantly improve PSNR and SSIM for both ITS and OTS models beyond the initial 20 epochs for ITS and 10 epochs for OTS, where M=0.2 was identified as optimal. The PSNR variation curves for both training processes (Figure 4) exhibit a similar trend: a steady increase in the first half of training, followed by a plateau, indicating potential early saturation in local optima.

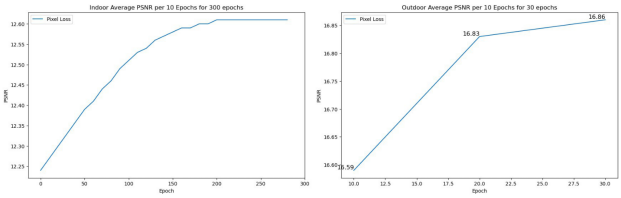


Figure 4: Combined PSNR variation curves for ITS and OTS training processes under Nesterov Momentum=0.2. **Left**: ITS Model trained for 300 epochs. **Right**: OTS Model trained for 30 epochs.

Nesterov Momentum's anticipatory updates may misalign with IRNeXt's structure and modules like MSM and LAM, which is crucial for scaling and modulation (Eq. 17). Its predictive nature might misjudge image degradation complexities, leading to premature local optima convergence. IRNeXt's dual-domain loss, merging spatial and frequency dimensions, complicates optimization. A static momentum (M=0.2) may restrict adaptability, affecting fine-tuning and post-initial improvement across domains.

Baselines (M=0) and potentially alternatively optimized original versions outdo our Nesterov-adapted IRNeXt, underscoring the necessity for adaptive momentum that suits IRNeXt's unique architecture and the nuances of image restoration, ensuring uniform learning and optimal problem-solving across scales and domains.

## Conclusion

In this report, we aim to enhance the model presented in the paper "IRNeXt: Rethinking Convolutional Network Design for Image Restoration", by adding Weight Decay, conducting Mix Up and changing its optimizer to Nesterov Momentum Optimizer. Though all 3 methods did not achieve better results than that stated in the paper, weight decay achieves the highest results where its performance exceeded the author's model when ran in our environment. Perhaps a different weight decay value, employing different data augmentation or segregating indoor and outdoor model training instead of multitasking with mixup, optimize the Nesterov Momentum value at smaller steps instead of using steps of 0.2 or exploring other optimizers can give better results than the paper.

## References


[1] Yuning Cui, Wenqi Ren, Sining Yang, Xiaochun Cao, and Alois Knoll. Irnext: Rethinking convolutional network design for image restoration. 2023.

[2] Anders Krogh and John A. Hertz. A simple weight decay can improve generalization. *Advances in Neural Information Processing Systems*, 4:950–957, 1992.

[3] Leslie N. Smith. A disciplined approach to neural network hyper-parameters: Part 1 – learning rate, batch size, momentum, and weight decay. *arXiv*, abs/1803.09820, 2018.

[4] Ian Goodfellow, Yoshua Bengio, and Aaron Courville. *Deep Learning*. MIT Press, 2016.

[5] Sebastian Ruder. An overview of gradient descent optimization algorithms. *arXiv*, abs/1609.04747, 2016.

[6] Chiyuan Zhang, Samy Bengio, Moritz Hardt, Benjamin Recht, and Oriol Vinyals. Understanding deep learning requires rethinking generalization. *Communications of the ACM*, 64(3):107–115, 2021.

[7] Tong He, Zhi Zhang, Hang Zhang, Zhongyue Zhang, Junyuan Xie, and Mu Li. Bag of tricks for image classification with convolutional neural networks. *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 558–567, 2019.

[8] Diederik P. Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *International Conference on Learning Representations (ICLR)*, 2014.

[9] Ilya Loshchilov and Frank Hutter. Fixing weight decay regularization in adam. *arXiv*, abs/1711.05101, 2017.

[10] Ilya Loshchilov and Frank Hutter. Decoupled weight decay regularization. *International Conference on Learning Representations (ICLR)*, 2019.

[11] Sashank J. Reddi, Satyen Kale, and Sanjiv Kumar. On the convergence of adam and beyond. *International Conference on Learning Representations (ICLR)*, 2019.

[12] Pavel Izmailov, Dmitrii Podoprikhin, Timur Garipov, Dmitry Vetrov, and Andrew Gordon Wilson. Averaging weights leads to wider optima and better generalization. *Uncertainty in Artificial Intelligence (UAI)*, 2018.

[13] Tianyi Zhang, Felix Wu, Arzoo Katiyar, Kilian Q. Weinberger, and Yoav Artzi. Revisiting few-sample bert fine-tuning. *International Conference on Learning Representations (ICLR)*, 2020.

[14] Douglas M. Hawkins. The problem of overfitting. *Journal of Chemical Information and Computer Sciences*, 44(1):1–12, 2004.

[15] Nitish Srivastava, Geoffrey Hinton, Alex Krizhevsky, Ilya Sutskever, and Ruslan Salakhutdinov. Dropout: A simple way to prevent neural networks from overfitting. *The Journal of Machine Learning Research*, 15(1):1929–1958, 2014.

[16] Xie Ying. An overview of overfitting and its solutions. In *Journal of Physics: Conference Series*, volume 1168, page 022022. IOP Publishing, 2019.

[17] Ashia C. Wilson, Rebecca Roelofs, Mitchell Stern, Nathan Srebro, and Benjamin Recht. The marginal value of adaptive gradient methods in machine learning. *Advances in Neural Information Processing Systems*, 30, 2017.

[18] Hongyi Zhang, Moustapha Cisse, Yann N. Dauphin, and David Lopez-Paz. mixup: Beyond empirical risk minimization. *International Conference on Learning Representations (ICLR)*, 2018.

[19] Yuji Tokozume, Yoshitaka Ushiku, and Tatsuya Harada. Between-class learning for image classification. *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 5486–5494, 2018.

[20] Dan Hendrycks, Norman Mu, Ekin D. Cubuk, Barret Zoph, Justin Gilmer, and Balaji Lakshminarayanan. Augmix: A simple data processing method to improve robustness and uncertainty. *International Conference on Learning Representations (ICLR)*, 2019.

[21] Sunil Thulasidasan, Tanmoy Bhattacharya, Janardhan Rao Doppa, Garrett Kenyon, and Louis N. C. Feldman. On mixup training: Improved calibration and predictive uncertainty for deep neural networks. *NeurIPS*, 2019.

[22] Hongyu Guo, Yongyi Mao, and Richong Zhang. Mixup as locally linear out-of-distribution regularization. *AAAI Conference on Artificial Intelligence*, 2019.

[23] Vikas Verma, Alex Lamb, Juho Kannala, Yoshua Bengio, and David Lopez-Paz. Manifold mixup: Better representations by interpolating hidden states. *International Conference on Machine Learning (ICML)*, 2019.

[24] Boyi Li, Xiulian Peng, Zhangyang Wang, Jizheng Xu, and Dan Feng. Reside: A benchmark for single image dehazing. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2018.

[25] Hiroshi Inoue. Data augmentation by pairing samples for images classification. *arXiv*, abs/1801.02929, 2018.

[26] Boyi Li, Xiulian Peng, Zhangyang Wang, Jizheng Xu, and Dan Feng. Reside: A benchmark for single image dehazing. *IEEE Transactions on Image Processing*, 28(1):4926–4940, 2018.

[27] Rich Caruana. Multitask learning. In *Machine Learning*, volume 28, pages 41–75, 1997.

[28] Sylvestre-Alvise Rebuffi, Hakan Bilen, and Andrea Vedaldi. Learning multiple visual domains with residual adapters. In *Advances in Neural Information Processing Systems*, pages 506–516, 2017.

[29] Ilya Sutskever, James Martens, George E. Dahl, and Geoffrey E. Hinton. On the importance of initialization and momentum in deep learning. In *International Conference on Machine Learning (ICML)*, pages 1139–1147, 2013.

[30] Yurii Nesterov. A method for solving the convex programming problem with convergence rate $o(1/k^2)$. *Doklady AN USSR*, 269:543–547, 1983.

[31] Aleksandar Botev, Guy Lever, and David Barber. Nesterov's accelerated gradient and momentum as approximations to regularised update descent. *International Joint Conference on Neural Networks (IJCNN)*, pages 1899–1903, 2017.