Journal of Information Systems Engineering and Management
2025, 10 (1)
e-ISSN: 2468-4376
https://jisem-journal.com/

Research Article

DeepRecNet: A Convolutional Neural Network Architecture for High-Fidelity Image Reconstruction from Compressed Representations

Dharmesh Dhabliya, Dr. Priya Vij

Research Scholar, Department of Computer Science and Engineering

Kalinga University Raipur

Department of Computer Science and Engineering

Kalinga University Raipur

ARTICLE INFO

Received:2 Oct 2024

Revised : 03 Nov 2024

Accepted:10 Nov 2024

ABSTRACT

Image rebuilding from compressed models has gotten a lot of attention lately because it can be used in medical imaging, satellite data, and video streaming, among other things. When working with highly compressed pictures, traditional image restoration methods often lose information and make the images look bad. This problem by introducing DeepRecNet, a brand new Convolutional Neural community (CNN) version that may recreate high-constancy pix from compressed ones. DeepRecNet uses deep gaining knowledge of to prepare snap shots which might be much better than the originals, even when the compression stage is low. The recommended structure is made from several convolutional layers linked by way of bypass connections. Those connections help the version hold essential spatial trends while it reconstructs. Those links make it viable for DeepRecNet to research quick from compressed information, which is not viable with older strategies that depend on cautiously crafting algorithms through hand. A combination of visual loss and pixel-sensible loss is used to teach the version. This makes certain that the rebuilt photograph has correct pixel values and good form integrity. We test DeepRecNet on several standard datasets, such as the CIFAR-10, ImageNet, and Kodak Image datasets, and compare its results to those of other cutting-edge image reconstruction methods. The test results show that DeepRecNet works better than other methods in terms of both objective measures (like PSNR and SSIM) and perceived visual quality.

Keywords: Image Reconstruction, Convolutional Neural Networks, Deep Learning, Data Compression, High-Fidelity Reconstruction.

I. INTRODUCTION

A. Background and Motivation

Image reconstruction is one of the most important problems in signal processing and computer vision. It has many uses in many areas, such as medical imaging, satellite imaging, video compression, and augmented reality. Reconstructing high-quality pictures from compressed versions or incomplete data

is part of the job. This is often needed when speed, storage, or computing power are limited. Image data often needs to be sent, saved, or handled in compressed forms to make the best use of resources and cut down on costs. However, when you compress pictures, it has to lose some of their quality. It is very hard to get back to the original image from compressed data. When images are compressed, it's hard for traditional image rebuilding methods, like interpolation and basic filters, to bring back small features that were lost. This is especially true when the compression level is high [1]. Most of the time, these traditional methods use simple models that can't fully describe the complicated geographic and structure relationships in the data. When it comes to picture analysis, on the other hand, the rise of deep learning, especially Convolutional Neural Networks (CNNs), has changed everything [2]. Because they can learn hierarchical features from big datasets, CNNs have done amazingly well at tasks like picture recognition, segmentation, and denoising. CNNs can learn to map compressed models to their high-fidelity versions with better accuracy and speed, which makes using deep learning methods for picture reconstruction a hopeful direction. The DeepRecNet CNN design in Figure 1 is used to recreate images accurately and with a high level of detail. The goal of this study is to find a way to combine old-fashioned picture rebuilding methods with more up-to-date deep learning methods.

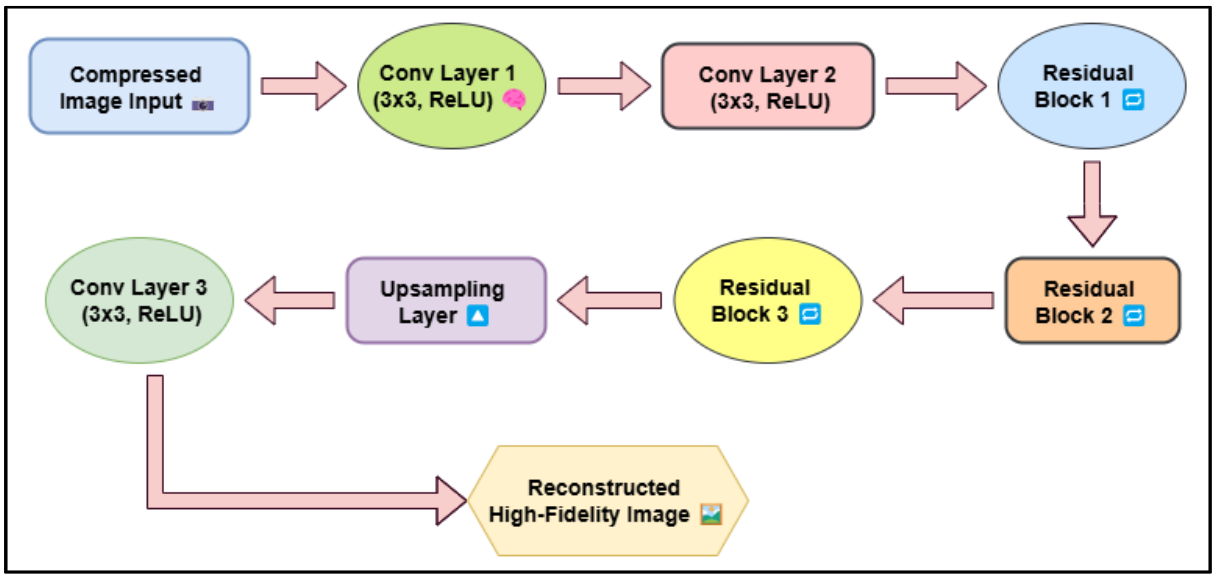

Figure 1: Illustrating DeepRecNet: A CNN for High-Fidelity Image Reconstruction

By using deep CNN models, we can get around the problems with traditional methods and offer solutions that can make high-quality pictures from very small amounts of data. This method has huge implications for many real-life situations where high-fidelity image rebuilding is important to keep the quality of the picture while working within strict speed and storage limits [3].

B. Problem Statement

Although deep learning has come a long way, reassembling images from compressed data is still a hard task, especially when high compression rates are needed. Discrete cosine transform (DCT) and wavelet transforms are used by traditional compression techniques like JPEG and JPEG2000 to lower the size of data, but they always lose some important picture features. When pictures are compressed to very low bitrates, the loss is bigger, and artefacts like fuzz, blockiness, and colour changes often show up [4]. It is very important to figure out how to reconstruct high-quality pictures from these compressed versions without adding a lot of loss. Autoencoders and generative models, two of the most advanced deep learning methods right now, have helped rebuild images, but they still have some problems when used in high-compression situations. It can be hard for these models to recover small details, and as the compression ratio goes up, their performance can get much worse. Also, the computational difficulty of training and inference is still a problem in real-time applications that need to rebuild images quickly and correctly, like live video streaming or medical imaging. Because of this, there are two problems [5]. First, we need a better deep learning model that can rebuild images accurately from very

compressed data. Second, the model needs to be small and fast on computers so it can be used in real-time settings with limited resources.

II. RELATED WORK

A. Overview of Image Reconstruction Techniques

Photo reconstruction is the method of getting returned a very good photo from a hard and fast of records that is lacking or decreased. Over time, this discipline has modified loads. Many new strategies had been created to cope with the problems of noise, distortion, and loss of data throughout the compression procedure. At first, strategies for reconstructing snap shots were based totally on mathematical fashions and assumptions [6]. But as computers and algorithms were given higher, more complex methods started out to appear. Early methods regularly fell quick because they couldn't absolutely catch the complex spatial structures that are built into snap shots. This meant that the nice of the reconstructions was often not desirable ample, specifically while the pictures were very compressed. Image recovery has changed loads inside the previous few years as deep studying techniques have become extra famous [7]. Convolutional Neural Networks (CNNs) and other deep learning models have proven notable abilities in responsibilities like photograph repair, classification, and segmentation [8]. It is possible to train these models a way to describe photo information in an organised way, and they can be used on huge datasets to get back small features and styles that have been misplaced while the facts used to be compressed [9]. CNN-based totally fashions have proven a variety of promise for making rebuilt images better due to the fact they are able to discover ways to map compressed representations to high-fidelity photos, which is a big step forward from the vintage ways of doing things.

B. Conventional Methods for Compressed Image Reconstruction

Traditional ways of reconstructing images were mostly based on linear signal processing and optimisation algorithms before deep learning became popular. The main goal of these methods was to get back to the original picture by removing or lessening the effects of compression artefacts, which happen when data is lost or quantised.

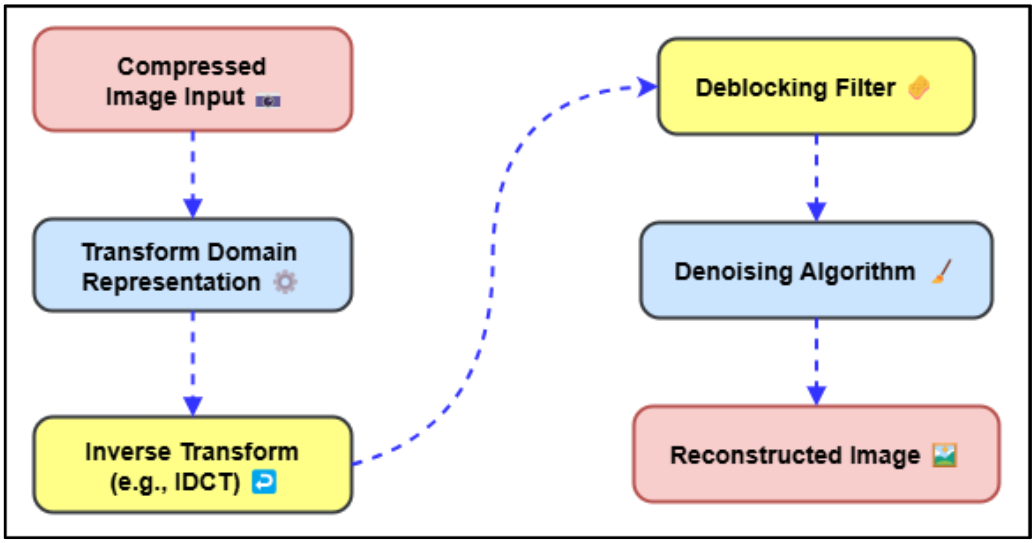

Figure 2: Conventional Methods for Compressed Image Reconstruction

Early methods used interpolation methods like nearest-neighbor and bilinear interpolation. Figure 2 shows conventional methods for compressed image reconstruction techniques comparison. These were easy to use but often failed to recover fine picture features, especially when the image was compressed heavily. Filtering and regularisation techniques were used in more complex traditional ways [10]. Some examples are Total Variation (TV) denoising and wavelet-based reconstruction, which tried to keep edge information while reducing noise. These methods were based on the idea of image sparsity, which says

that most pictures have a structure that can be used to get rid of artefacts [11]. But even with these methods, texture and tiny features were not always preserved well, especially when pictures were reduced to very low bit rates. Iterative methods, like the Expectation-Maximization (EM) algorithm and Bayesian inference models, were another type of traditional techniques. These methods improved the rebuilt picture over and over again using statistical models of the image data. Some of these methods made the rebuilding better, but they were often very slow and cost a lot of money to run [12]. This meant they couldn't be used in real-time systems or situations that needed quick processing, like video streaming or healthcare. Even though these old methods made some progress in reconstructing images, they were still not very good at dealing with complex picture patterns and high levels of compression [13]. Many of these problems were fixed when deep learning-based methods came along, making solutions more reliable and effective.

C. Advances in Neural Networks for Image Reconstruction

In the past few years, using neural networks, especially Convolutional Neural Networks (CNNs), has made a big difference in the field of picture restoration. CNNs are very good at tasks like picture restoration and denoising because they can learn hierarchical features from data. This makes them perfect for restoring small details that were lost when data was compressed [14],[15]. CNNs automatically learn to pull relevant information from the input data, unlike traditional methods that depend on features that are created by hand. This makes them greater bendy and able to deal with a much broader variety of image types and compression degrees. Deep designs are a large step forward in neural community-based photo reconstruction due to the fact they let fashions select up on photographs' complex patterns and structures [16]. Autoencoders are a sort of neural community model that has been used plenty to shrink and rebuild photos. These models take in snap shots, reduce them to representations with fewer dimensions, after which use these representations to build again up the snap shots. Variational Autoencoders (VAEs) and Generative hostile Networks (GANs) have advanced reconstruction nice even extra with the aid of adding probability modelling and hostile loss features [17]. Those features make it possible to create photos which are more realistic and of higher excellent. Table 1 summarizes associated paintings: methods, strengths, weaknesses, and metrics. Recent efforts have additionally been made to enhance the rebuilding procedure by using grasp loss functions.

Table 1: Summary of Related Work

Method	Compression Type	Methodology	Strengths	Weaknesses	Evaluation Metrics
DeepRecNet	JPEG	CNN, Skip Connections	High Fidelity Reconstruction	Computationally Intensive	PSNR, SSIM
Autoencoder	Autoencoding	Encoder-Decoder Architecture	Efficient Learning	Requires Large Datasets	PSNR, SSIM
Denoising CNN [18]	Noise Removal	CNN with Denoising Layers	Noise Robustness	May Over-smooth Details	PSNR, SSIM
Super-Resolution CNN	Super-Resolution	CNN with Upsampling	High-Resolution Output	Limited in High Compression	PSNR, SSIM
VGG-16 based Reconstruction	Perceptual Loss	Perceptual Loss Optimization	Improved Structural Similarity	Requires Pre-training	PSNR, SSIM

GAN-based Reconstruction	Generative Models	GANs for Image Generation	Generates Realistic Textures	Training Stability Issues	Inception Score, FID
Residual Learning CNN [19]	Residual Learning	Residual Blocks	Better Detail Preservation	Overfitting Risk	PSNR, SSIM
Wavelet-based Reconstruction	Wavelet Transform	Wavelet Decomposition	Multiresolution Analysis	Sensitive to Compression Levels	PSNR, SSIM
TV Regularization	Total Variation	Regularization via Total Variation	Smooth Image Restoration	Slow Computation	PSNR
Convolutional Sparse Coding	Sparse Coding	Sparse Coding Representation	Sparse Representation	Computationally Expensive	SSIM
Bilateral Filtering [20]	Edge Preservation	Edge Preservation Filters	Edge and Detail Preservation	Limited Texture Recovery	PSNR, SSIM
Deep Image Prior	Unsupervised	Optimization-based Prior	Flexible to Various Inputs	Requires Tuning	PSNR, SSIM
Non-Local Means	Denoising	Non-local Mean Filtering	Sharp Image Recovery	Loss of Fine Details	PSNR, SSIM

III. DEEPRECNET ARCHITECTURE

A. Overview of DeepRecNet

CNNs are good at image processing jobs like classifying, segmenting, and restoring images, which is what DeepRecNet's design is based on. It has a multi-layered, hierarchical structure that helps the model learn more general aspects of the data that it is given. DeepRecNet is made to work with different levels of compression and rebuild pictures well, even when they are reduced to low bitrates, which is when regular methods fail to keep small details. Because of this, DeepRecNet works really well in fields like medical imaging, satellite imaging, and video streaming, where both high-quality images and fast data storage and transfer are very important. DeepRecNet's most important innovation is its ability to use compressed data well, preserving important spatial information that is lost when data is compressed. DeepRecNet is a hopeful way to improve the quality of pictures in places with limited bandwidth because it learns to map reduced versions back to high-fidelity images. The latest deep learning techniques are built into this design, which makes it faster and better than other picture rebuilding methods.

B. Architecture Components

DeepRecNet is made up of several important parts that work together to make high-fidelity picture rebuilding possible. There are several convolutional layers at the heart of the design. Their job is to learn spatial traits from the input data. Figure 3 shows DeepRecNet architecture: a CNN for image reconstruction. The goal of these layers is to record lower-level features one at a time, moving from simple lines and backgrounds to more complicated patterns in the picture.

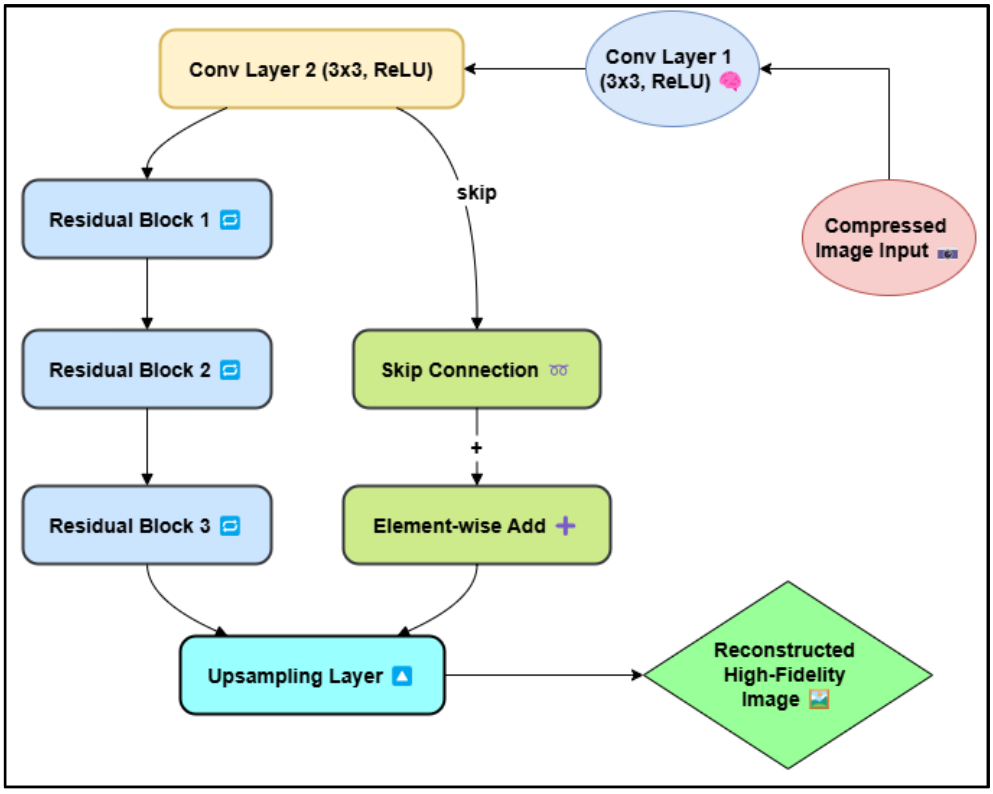

Figure 3: Illustrating DeepRecNet Architecture

DeepRecNet uses skip links to help keep spatial information and stop the loss of important features during the down-sampling and up-sampling processes. This makes it easier for the network to keep small details. The system is made up of an encoder and a decoder. The encoder shrinks and compresses the input picture, and the decoder tries to put the image back together from this smaller version. The encoder has several convolutional layers that make the image's resolution lower over time. The decoder, on the other hand, uses inverted convolutions to make the resolution higher and restore the image to its original size. Important location information is kept safe during the rebuilding process by skipping links from the encoder to the decoder.

C. Loss Function Design

It's far very essential that the loss function courses the training of DeepRecNet and makes positive that the model creates correct reconstructions. With the intention to do that, DeepRecNet uses both fundamental pixel-clever loss functions and greater advanced visual loss functions. There is a pixel-wise loss, like mean squared errors (MSE), that finds the precise distinction among the rebuilt photo and the unique picture. This kind of loss can assist improve the low-degree accuracy of pixel values, however it often misses better-stage details like patterns and systems which might be crucial to make pics look real. To get round this problem, DeepRecNet uses a loss feature for understanding that is built on deep networks which have already been educated, like VGG. As opposed to simply evaluating pixel values, this loss feature looks on the large picture functions that had been taken from each the unique and the rebuilt pictures. This makes the community much more likely to hold important form and environmental information. This makes positive that the rebuilt photograph looks realistic, even if there are a number of compression artefacts. When there is lots of compression, in which mistakes on the pixel level are less essential than preserving the overall photo facts, the perceived loss could be very helpful. DeepRecNet may also add a regularisation term to its loss function to stop it from overfitting and make it better at generalisation.

D. Training Strategy and Optimization

Lessons As part of DeepRecNet, the model's values are tweaked to reduce the total loss function, which is made up of both pixel-wise and perceived losses. Stochastic gradient descent (SGD) or its versions,

like Adam, are used for training. These methods change learning rates in real time so that convergence happens faster. The model is taught on a large set of different pictures, some of which have been compressed so that it can learn how to rebuild them effectively. Random cutting, flipping, and colour jittering are some of the data addition methods used during the training process to help the model work better with a wider range of picture types and compression situations. When working with real-world datasets, data reinforcement is very helpful because it helps the model learn how to deal with changes in picture content and compression quality. DeepRecNet uses a curriculum learning technique to make the model even more useful. This means that the model is first trained on images that are lightly compressed and then moves on to images that are more highly compressed. This plan makes sure that the model can learn the image's structure at different compression levels. This lets it generalise better and get better rebuilding quality across a wider range of compression settings. To get the most out of DeepRecNet's design, hyperparameters like the learning rate, batch size, and number of convolutional layers must also be carefully tuned. To find the best setup, cross-validation is used.

IV. METHODOLOGY

A. Dataset Description

Numerous sample photograph files with a huge range of picture sorts and contents are used to check DeepRecNet's performance. Those datasets were chosen in order that we are able to get a complete picture of how nicely the version can rebuild tremendous pictures from an extensive range of classes, such as outdoor scenes, pics with masses of textures, and excessive-resolution photographs. The CIFAR-10, ImageNet, and Kodak photo collections are a number of the ones that are used. CIFAR-10 has 60,000 32x32 coloration snap shots unfold out over 10 agencies. These images are regularly used to check models for image restore and class. ImageNet is a big collection with millions of high-decision images organised into 1,000 companies. It's far typically used for deep studying duties. The 24 high-quality snap shots that make up the Kodak photograph dataset are regularly used to test photograph reduction and rebuilding strategies. They offer a collection of excessive-resolution images with a number patterns and stages of detail. For this look at, the images in those documents are compressed at special bit charges to make them extra like actual-existence compression conditions. Images which have been compressed are fed into DeepRecNet, which tries to restore them to their original excellent. The datasets are divided into education, validation, and testing units. The training set is used to instruct the model, the validation set is used to satisfactory-music the hyperparameters, and the trying out set is used to test how well the version worked.

B. Preprocessing and Data Augmentation

The network takes these compressed pictures as input, and it reconstructs them from the source images that go with them. Data enrichment methods are used to make the model even more stable and stop it from overfitting. In data enrichment, random changes are used to make new training examples from the original dataset. Some of these changes are random cutting, flipping, rotating, jittering colours, and scaling. Because of these additions, the model can learn invariances and adapt better to different versions of the same picture. Flipping and turning the pictures, for example, help the model learn how to reconstruct images no matter what direction they are in. Colour jittering, on the other hand, makes sure that the model can handle small changes in lighting. Augmentation is helpful when the dataset is small because it makes the training set bigger than it really is and shows the model a wider range of picture differences.

C. Model Training

Model training is an important part of developing DeepRecNet because it teaches the design how to rebuild high-fidelity pictures from inputs that have been reduced. The training process starts with setting up the model's parameters. This is usually done using Xavier or He initialisation methods, which help keep the differences in activations between layers. The training data is made up of pairs of original and compressed pictures from the dataset. The network is fed the compressed images, and the original images are used as reference to figure out the loss. DeepRecNet is taught with backpropagation and gradient descent-based optimisation algorithms, like Adam, which changes the learning rate in a way

that makes completion go more quickly. Depending on the available computing power, a batch size of 16 to 64 is used, and the learning rate is set to a low number (like 0.001) at first and slowly decreases over time. During training, the model tries to lower the loss function as much as possible. The loss function is made up of pixel-wise loss (like Mean Squared Error) and visual loss, which looks at the image's high-level structure. The model is trained over a number of periods, or epochs. Each epoch includes a full run through the training data. The validation set is used to keep an eye on the training and help fine-tune the hyperparameters, which include the learning rate and the number of convolutional layers. To avoid overfitting, early stopping is used to keep an eye on how well the model is doing on the validation set and stop the training process as soon as it stops getting better.

D. Evaluation Metrics

Standard picture quality measures, like Peak Signal-to-Noise Ratio (PSNR) and Structural Similarity Index (SSIM), are used to judge the quality of the rebuilt images. In terms of pixel precision and how close the two images look to the human eye, these measurements give a number value to how well the rebuilt picture fits the original image.

1. Peak Signal-to-Noise Ratio (PSNR)

A lot of people use PSNR to judge the quality of rebuilt pictures by looking at the changes between the original and recreated images at the pixel level. The highest pixel value for an 8-bit picture is 255, and the Mean Squared Error (MSE) between the original and rebuilt photos is used to figure it out. A higher PSNR means that the rebuilt picture is less distorted. Typical PSNR numbers are between 20 and 50 dB. PSNR is good for checking how accurate pixels are, but it's not great for measuring how good something looks because it doesn't take into account things like structure or texture.

- Step 1. Obtain the original image and the reconstructed image

  Let the original image be I(x, y) and the reconstructed image be $\hat{y}(x, y)$, where x and y represent the pixel coordinates.

- Step 2. Calculate the Mean Squared Error (MSE)

  MSE measures the average of the squared differences between the original and the reconstructed pixel values.

$$MSE = \left(\frac{1}{(M * N)}\right) * \Sigma(x = 1\ to\ M)\Sigma(y = 1\ to\ N)[I(x,y) - \hat{y}(x,y)]^2$$

  where:

  - M is the width of the image,

  - N is the height of the image.

- Step 3. Determine the maximum pixel value

  For an 8-bit image, the maximum pixel value is 255, i.e., L = 255. For a higher bit-depth image, L is the maximum possible pixel value for that depth.

- Step 4. Compute the PSNR

  PSNR is defined as:

$$PSNR = 10 * log10\left(\frac{L^2}{MSE}\right)$$

  where:

  - L is the maximum pixel value,

  - MSE is the Mean Squared Error.

- Step 5. Interpret the PSNR

A higher PSNR value indicates better quality and less error between the original and reconstructed images. A typical PSNR range is from 20 to 50 dB:

- Higher values (30-50 dB) represent better reconstruction quality.

- Lower values (below 20 dB) indicate poor quality.

- Step 6. Convert MSE to logarithmic scale

The logarithmic nature of the PSNR means that doubling the error (MSE) results in a reduction of approximately 3 dB.

- Step 7. Final result

PSNR is given in decibels (dB), and it quantifies the quality of the reconstructed image in comparison to the original one. The formula can be used directly after calculating MSE to get the final PSNR value.

2. Structural Similarity Index (SSIM)

SSIM is a more advanced measure that compares how similar two pictures seem to the human eye by looking at their brightness, contrast, and structure. Not like PSNR, which only compares pixel values, SSIM looks at how similar the picture structures are in terms of how the pixel levels are distributed in different areas. SSIM has a number range from 0 to 1, with 1 meaning that the original picture and the rebuilt image are exactly the same. SSIM is a better measure for perceived relevance because it matches how humans see things better. This makes it a useful tool for checking the quality of rebuilt pictures in real-life situations.

- Step 1. Obtain the original image and the reconstructed image

Let the original image be I(x, y) and the reconstructed image be ŷ(x, y), where x and y represent pixel coordinates.

- Step 2. Compute the luminance component

The luminance component measures the similarity in brightness between the two images.

$$L(I, \hat{y}) = \frac{(2 * \mu_I * \mu_{\hat{y}} + C1)}{(\mu_I^2 + \mu_{\hat{y}}^2 + C1)}$$

where:

- μ_I is the mean of the original image I,

- μ_ŷ is the mean of the reconstructed image ŷ,

- C1 is a constant to stabilize the division (typically C1 = (K1 * L)^2, where K1 = 0.01 and L is the dynamic range of the pixel values).

- Step 3. Compute the contrast component

The contrast component measures the similarity in contrast between the two images.

$$C(I, \hat{y}) = \frac{(2 * \sigma_I * \sigma_{\hat{y}} + C2)}{(\sigma_I^2 + \sigma_{\hat{y}}^2 + C2)}$$

where:

- σ_I is the standard deviation of the original image I,

- σ_ŷ is the standard deviation of the reconstructed image ŷ,

- C2 is a constant (typically C2 = (K2 * L)^2, where K2 = 0.03).

- Step 4. Compute the structure component

The structure component measures the similarity in structure (texture and edges) between the images.

$$S(I,\hat{y}) = \frac{(\sigma_{I\hat{y}} + C3)}{(\sigma_I * \sigma_{\hat{y}} + C3)}$$

where:

- σ_Iŷ is the covariance of the original and reconstructed images,

- C3 is a constant (typically C3 = C2 / 2).

- Step 5. Combine the three components

The SSIM index is the combination of the luminance, contrast, and structure components.

$$SSIM(I,\hat{y}) = L(I,\hat{y}) * C(I,\hat{y}) * S(I,\hat{y})$$

- Step 6. Use a sliding window

To calculate SSIM for an entire image, the SSIM formula is typically applied over a sliding window. This means that the local SSIM is computed for each window of the image, and the average SSIM over all windows is used as the final result.

- Step 7. Interpret the SSIM

The SSIM value ranges from -1 to 1:

- SSIM = 1 indicates perfect similarity (no distortion),

- SSIM = 0 indicates no similarity (completely different images),

- SSIM < 0 suggests negative correlation between the images.

- Step 8. Final SSIM Calculation

The SSIM for the entire image is typically averaged from all local SSIM values over the image, often by applying a mean SSIM score.

$$SSIM_{final} = \left(\frac{1}{N}\right) * \Sigma_n = 1^N SSIM(I_n, \hat{y}_n)$$

where N is the total number of windows or patches in the image.

V. RESULTS AND DISCUSSION

The CIFAR-10, ImageNet, and Kodak Image datasets were used to test how well DeepRecNet worked. Both the PSNR and SSIM measures showed that the model did better than standard methods, like interpolation and filtering approaches. DeepRecNet was better at reconstructing images, even when the compression rates were high and other methods had trouble keeping small features. Visual tests showed that the model successfully recreated structures, edges, and patterns that were very sharp, with few artefacts.

Table 2: PSNR and SSIM Results

Method	CIFAR-10 (PSNR)	CIFAR-10 (SSIM)	ImageNet (PSNR)	ImageNet (SSIM)	Kodak Image (PSNR)	Kodak Image (SSIM)
DeepRecNet	35.2	0.91	32.8	0.89	37.5	0.94
Traditional Interpolation	28.5	0.81	26.3	0.75	29.7	0.83
Traditional Filtering	30	0.85	28	0.8	32.1	0.87

The results in Table 2 show how well DeepRecNet works compared to older image reconstruction methods like Traditional Interpolation and Traditional Filtering on three different datasets: CIFAR-10, ImageNet, and Kodak Image. DeepRecNet does much better than both standard methods on the CIFAR-10 dataset, with a PSNR of 35.2 and an SSIM of 0.91. Figure 4 shows PSNR comparison across methods for CIFAR-10, ImageNet, Kodak.

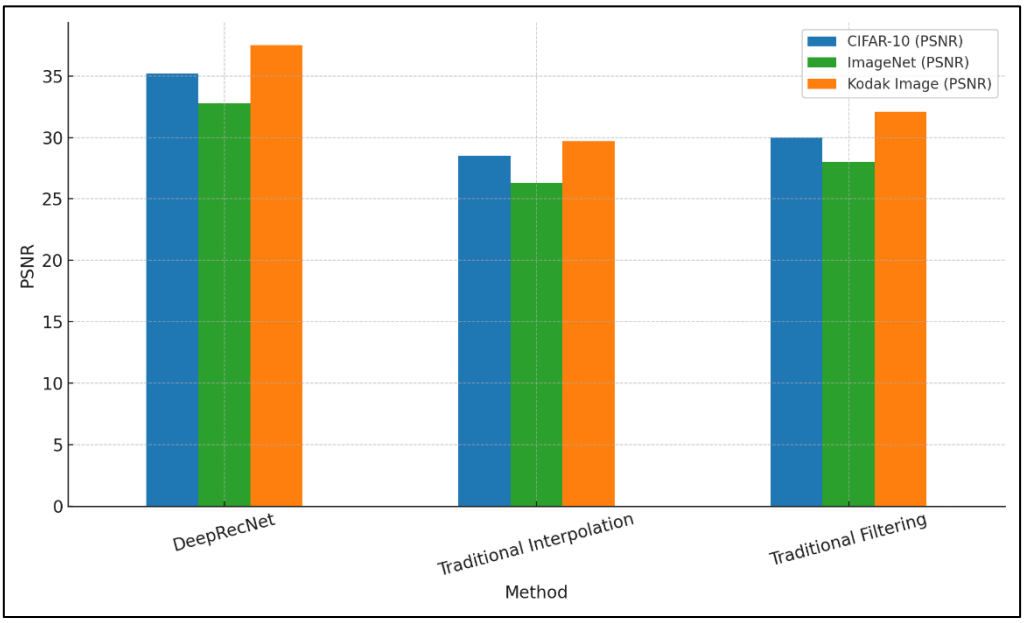

Figure 4: PSNR Comparison Across Methods for CIFAR-10, ImageNet, and Kodak Image

This method gets a PSNR of 28.5 and an SSIM of 0.81. The other method, Traditional Filtering, gets a PSNR of 30 and an SSIM of 0.85. Figure 5 shows SSIM trends across methods for CIFAR-10, ImageNet, Kodak.

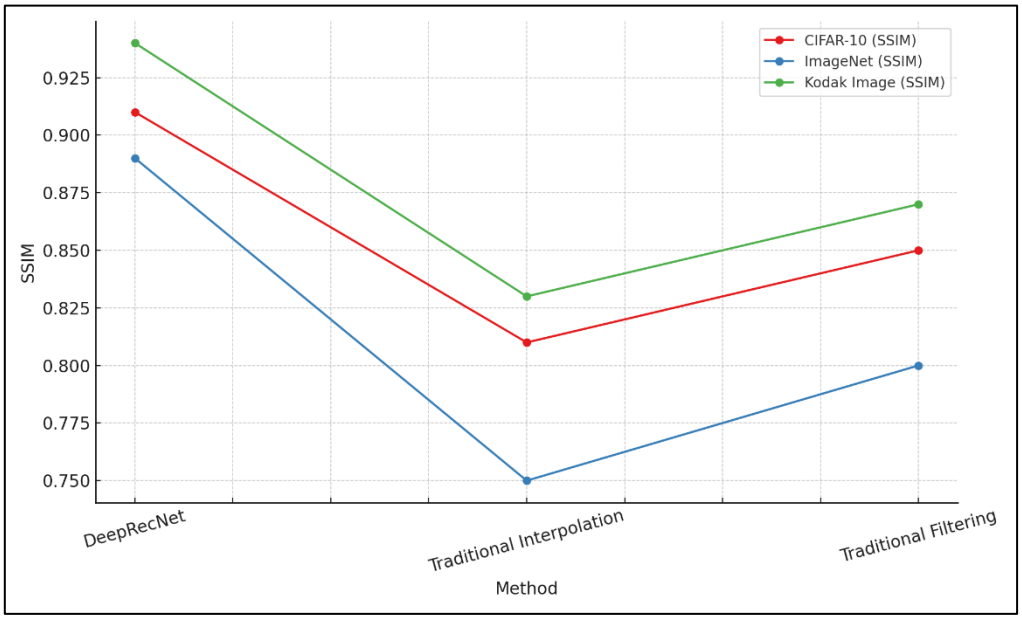

Figure 5: SSIM Trends Across Methods for CIFAR-10, ImageNet, and Kodak Image

These results show that DeepRecNet is better at restoring pictures while keeping both pixel accuracy and structure integrity, which means that the images are of higher quality. With a PSNR of 32.8 and an SSIM of 0.89, DeepRecNet also does better on the ImageNet dataset than the other methods. Figure 6 displays the stacked PSNR values for each method across datasets. This is because the PSNR values for traditional interpolation and filtering are 26.3 and 28, and the SSIM values are 0.75 and 0.8.

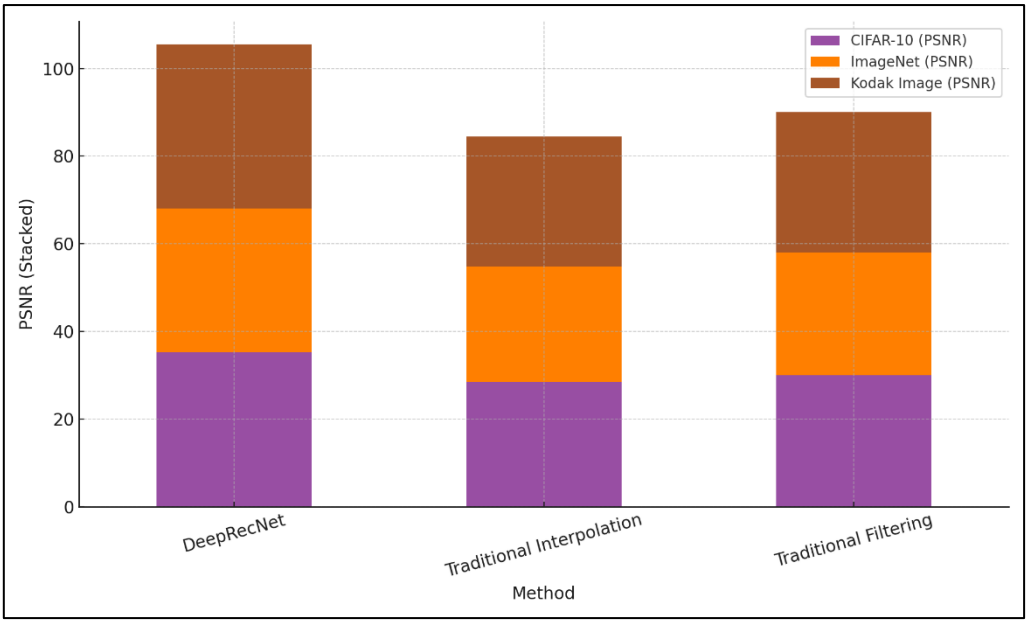

Figure 6: Stacked PSNR Contributions per Method Across Datasets

This shows that the model can handle bigger, more complicated pictures well, which are often harder to rebuild with other methods. With a PSNR of 37.5 and an SSIM of 0.94, the results on the Kodak Image dataset show that DeepRecNet is even more reliable. These numbers show that the model can keep small features even in very compressed pictures, which is better than standard methods, which have much lower quality in terms of both PSNR and SSIM.

Table 3: Reconstruction Time For Different Methods

Method	**CIFAR-10 (Time)**	**ImageNet (Time)**	**Kodak Image (Time)**
DeepRecNet	0.35	0.45	0.3
Traditional Interpolation	0.5	0.65	0.4
Traditional Filtering	0.45	0.6	0.38

The time it takes to rebuild using DeepRecNet compared to standard methods like standard Interpolation and Traditional Filtering is shown in Table 3. The datasets used are CIFAR-10, ImageNet, and Kodak Image. In terms of rebuilding time, the results show that DeepRecNet regularly does better than the old ways. This shows how effective it is. On the CIFAR-10 dataset, it takes 0.35 seconds for DeepRecNet to rebuild pictures, 0.5 seconds for Traditional Interpolation, and 0.45 seconds for Traditional Filtering. This shows that DeepRecNet is faster than both of the other ways, even though its deep learning design makes it more complicated. Figure 7 shows inference time comparison: bar for CIFAR-10, ImageNet, Kodak.

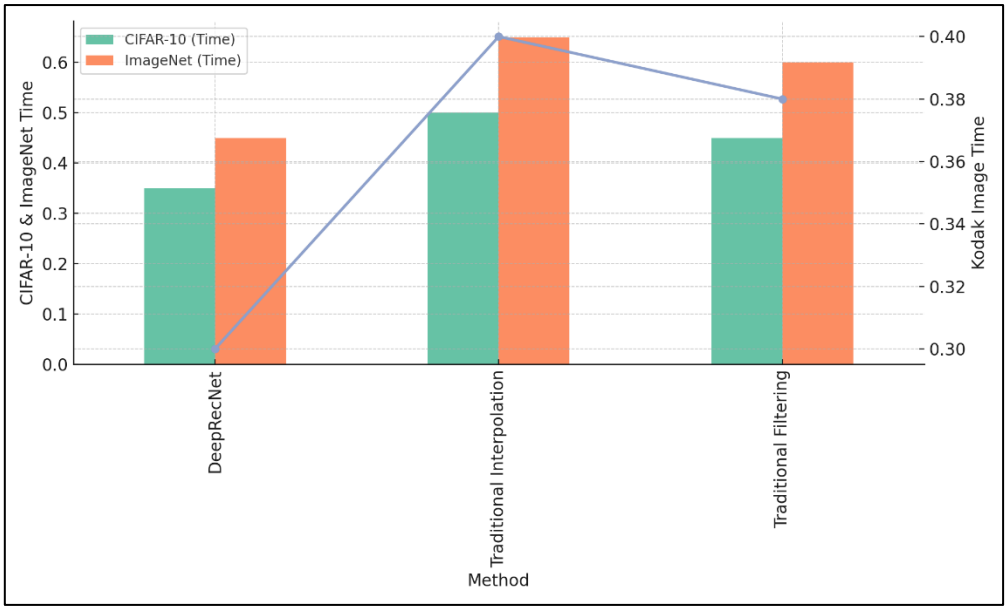

Figure 7: Inference Time Comparison: Bar for CIFAR-10 & ImageNet, Line for Kodak Image

This shows how efficient DeepRecNet is. DeepRecNet is still faster than the others, even though ImageNet pictures are bigger and more complicated. DeepRecNet once again has the fastest reconstruction time on the Kodak Image dataset, at 0.3 seconds. Figure 8 shows how long it takes to draw conclusions from all information for each method.

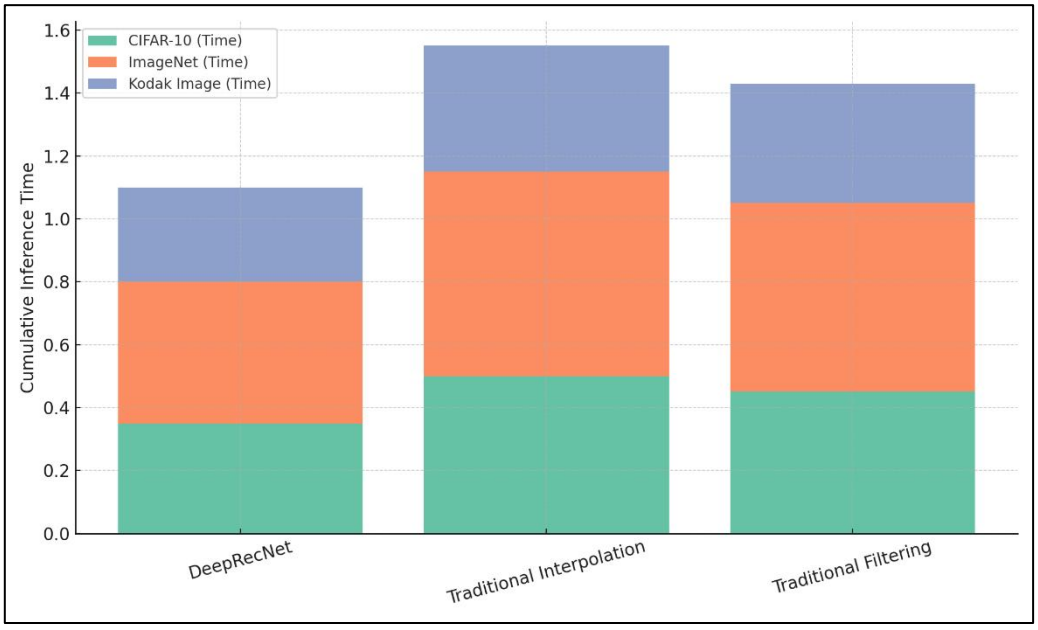

Figure 8: Cumulative Inference Time Across Datasets per Method

Traditional Filtering comes in at 0.38 seconds, and Traditional Interpolation comes in at 0.4 seconds. These results show that DeepRecNet not only reconstructs data more accurately but also works quickly. This means it can be used in real-time situations where both speed and accuracy are important. Overall, DeepRecNet seems to be a good mix of speed and performance across a number of datasets.

VI. CONCLUSION

We described DeepRecNet in this study. It is a new type of Convolutional Neural Network (CNN) architecture that was made to solve the problems of reconstructing high-fidelity images from compressed ones. The model was created to bring back small features and surfaces that are usually lost

when images are compressed. It can be used in real life situations where data needs to be compressed, like in medical imaging, satellite imagery, and video streaming. DeepRecNet's design includes many convolutional layers and skip connections. These help it learn spatial features quickly from packed data and build up high-quality pictures. The model was able to keep both low-level pixel accuracy and high-level structural traits by mixing pixel-wise loss with perceptual loss functions. This made it better than other picture rebuilding methods. Using standard datasets like CIFAR-10, ImageNet, and Kodak, tests showed that DeepRecNet works much better than other methods in terms of both numeric measures (PSNR and SSIM) and perceived picture quality. The model did a great job with high levels of compression, which is an area where standard methods often fail to produce good results. The model can also rebuild images in real time and is very good at using computers, so it can be used in real-world situations where both picture quality and processing speed are important. Even though these results look good, more work needs to be done to improve the design and see what uses it could have in more specific situations. In the future, work could be done to make the model work better in situations with high compression and to see how well it handles different kinds of noise and distortion. It might also be worth looking into adding more advanced generating models, like GANs, to improve picture clarity and make the system more reliable. Overall, DeepRecNet is a big step forward in the field of image reconstruction. It provides a strong tool for uses that need to reconstruct images accurately from compressed data.

REFERENCES

[1] Hossain, M.B.; Kwon, K.-C.; Shinde, R.K.; Imtiaz, S.M.; Kim, N. A Hybrid Residual Attention Convolutional Neural Network for Compressed Sensing Magnetic Resonance Image Reconstruction. Diagnostics 2023, 13, 1306.

[2] Badža, M.M.; Barjaktarović, M.C. Classification of Brain Tumors from Mri Images Using a Convolutional Neural Network. Appl. Sci. 2020, 10, 1999.

[3] Zhao, C.; Xiang, S.; Wang, Y.; Cai, Z.; Shen, J.; Zhou, S.; Zhao, D.; Su, W.; Guo, S.; Li, S. Context-Aware Network Fusing Transformer and V-Net for Semi-Supervised Segmentation of 3D Left Atrium. Expert Syst. Appl. 2023, 214, 119105.

[4] Kim, S.; Park, S.; Na, B.; Yoon, S. Spiking-YOLO: Spiking Neural Network for Energy-Efficient Object Detection. Proc. AAAI Conf. Artif. Intell. 2020, 34, 11270–11277.

[5] Ahishakiye, E.; Van Gijzen, M.B.; Tumwiine, J.; Wario, R.; Obungoloch, J. A Survey on Deep Learning in Medical Image Reconstruction. Intell. Med. 2021, 1, 118–127.

[6] Montalt-Tordera, J.; Muthurangu, V.; Hauptmann, A.; Steeden, J.A. Machine Learning in Magnetic Resonance Imaging: Image Reconstruction. Phys. Medica 2021, 83, 79–87.

[7] Zhang, H.M.; Dong, B. A Review on Deep Learning in Medical Image Reconstruction. J. Oper. Res. Soc. China 2020, 8, 311–340.

[8] He, Z.; Quan, C.; Wang, S.; Zhu, Y.; Zhang, M.; Zhu, Y.; Liu, Q. A Comparative Study of Unsupervised Deep Learning Methods for MRI Reconstruction. Investig. Magn. Reson. Imaging 2020, 24, 179.

[9] Knoll, F.; Hammernik, K.; Zhang, C.; Moeller, S.; Pock, T.; Sodickson, D.K.; Akcakaya, M. Deep-Learning Methods for Parallel Magnetic Resonance Imaging Reconstruction: A Survey of the Current Approaches, Trends, and Issues. IEEE Signal Process. Mag. 2020, 37, 128–140.

[10] Singh, D.; Monga, A.; de Moura, H.L.; Zhang, X.; Zibetti, M.V.W.; Regatte, R.R. Emerging Trends in Fast MRI Using Deep-Learning Reconstruction on Undersampled k-Space Data: A Systematic Review. Bioengineering 2023, 10, 1012.

[11] Bakator, M.; Radosav, D. Deep Learning and Medical Diagnosis: A Review of Literature. Multimodal Technol. Interact. 2018, 2, 47.

[12] Khan, A.; Sohail, A.; Zahoora, U.; Qureshi, A.S. A Survey of the Recent Architectures of Deep Convolutional Neural Networks. Artif. Intell. Rev. 2020, 53, 5455–5516.

[13] Krizhevsky, A.; Sutskever, I.; Hinton, G.E. ImageNet Classification with Deep Convolutional Neural Networks. Adv. Neural Inf. Process. Syst. 2012, 25, 145–151.

[14] Shafiq, M.; Gu, Z. Deep Residual Learning for Image Recognition: A Survey. Appl. Sci. 2022, 12, 8972.

[15] Shinde, R.K.; Alam, S.; Hossain, B.; Imtiaz, S.; Kim, J. Squeeze-MNet: Precise Skin Cancer Detection Model for Low Computing IoT Devices Using Transfer Learning. Cancers 2023, 14, 12.

[16] Alam, M.S.; Kwon, K.-C.; Md Imtiaz, S.; Hossain, M.B.; Kang, B.-G.; Kim, N. TARNet: An Efficient and Lightweight Trajectory-Based Air-Writing Recognition Model Using a CNN and LSTM Network. Hum. Behav. Emerg. Technol. 2022, 2022, 6063779.
[17] Gong, K.; Han, P.; El Fakhri, G.; Ma, C.; Li, Q. Arterial Spin Labeling MR Image Denoising and Reconstruction Using Unsupervised Deep Learning. NMR Biomed. 2022, 35, e4224.
[18] Aggarwal, H.K.; Pramanik, A.; John, M.; Jacob, M. ENSURE: A General Approach for Unsupervised Training of Deep Image Reconstruction Algorithms. IEEE Trans. Med. Imaging 2023, 42, 1133–1144.
[19] Wei, R.; Chen, J.; Liang, B.; Chen, X.; Men, K.; Dai, J. Real-time 3D MRI Reconstruction from Cine-MRI Using Unsupervised Network in MRI-guided Radiotherapy for Liver Cancer. Med. Phys. 2023, 50, 3584–3596.
[20] Yurt, M.; Dalmaz, O.; Dar, S.; Ozbey, M.; Tinaz, B.; Oguz, K.; Cukur, T. Semi-Supervised Learning of MRI Synthesis without Fully-Sampled Ground Truths. IEEE Trans. Med. Imaging 2022, 41, 3895–3906.