Bidirectional reservoir computing and self-attention models for image classification and inpainting

Phuc Khanh Nhi NGUYEN
npkhanhnhi@gmail.com

Abstract

In this project, we explore the integration of reservoir computing, specifically Echo State Networks (ESNs), into Deep Learning models to address two distinct tasks: image classification and inpainting. While ESNs are well-suited to processing data with an evolutionary direction, such as time series, images lack this temporal structure and emphasize spatial relationships instead. To bridge this gap, we propose a novel bidirectional computing structure that adapts ESNs for image-based tasks, enabling the extraction of more robust features. To validate this approach, we design a hybrid convolutional-ESN (Conv-ESN) feature extractor that serves both tasks. Additionally, for the inpainting task, we develop a specific decoder network employing a self-attention mechanism. Though a multi-task learning framework is not implemented, we investigate the transferability of the encoder learned for one task to the other. Experimental results on the STL-10 dataset demonstrate that the proposed bidirectional ESN structure significantly enhances classification accuracy. For the inpainting task, the reconstruction model achieves high PSNR and SSIM scores. However, transfer learning experiments reveal performance degradation when weights are transferred between tasks—particularly from the reconstruction to the classification model. UMAP visualizations further highlight discrepancies in the representations of native and weight-transferred classification models.

1 Introduction

Reservoir Computing (RC) has emerged as a powerful computational paradigm, finding applications in diverse fields such as bioinformatics, robotics, and environmental monitoring. In bioinformatics, RC models like Echo State Networks (ESNs) have been explored for analyzing biological systems, with applications ranging from bacterial reservoir computing [1] to multicellular systems that process information using cell-to-cell signaling [2]. Beyond bioinformatics, RC has also demonstrated effectiveness in robotics for sensor data processing [3], financial forecasting for analyzing temporal data [4], and speech recognition by extracting patterns in audio signals [5].

Recently, RC has gained traction in image-related tasks, leveraging its ability to adapt to complex spatial dynamics. For instance, RC has been successfully applied to image data as demonstrated in [6], [7]. These works highlight how RC models can be adapted to handle the spatial relationships in image data to predict the category of images. These advancements showcase the versatility of RC in addressing challenges in image processing. However, while ESNs excel at processing data with an evolutionary direction, such as time series, their adaptation to image-based tasks—where spatial relationships dominate—remains a challenging frontier. In this study, we propose a novel bidirectional computing structure to enable ESNs to extract robust features for image-based applications. Specifically, we address two distinct tasks: image classification, where accurate feature extraction is critical, and inpainting, which requires nuanced reconstruction to preserve structural and visual coherence.

To validate our approach, we design a hybrid convolutional-ESN (Conv-ESN) feature extractor for both tasks and develop a self-attention-based decoder for inpainting. Experimental results on the STL-

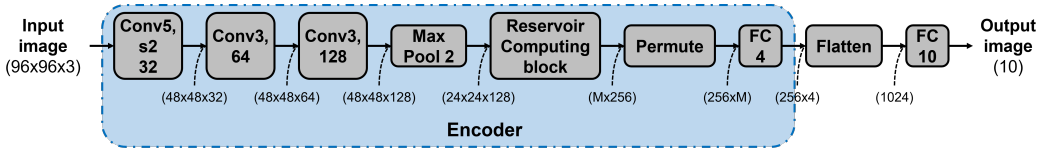


Figure 1: Architecture of the encoder and the classification model.

10 dataset demonstrate the bidirectional ESN structure's efficacy, achieving significant improvements in classification accuracy and state-of-the-art performance in inpainting as measured by PSNR and SSIM scores. However, transfer learning experiments highlight challenges, particularly when transferring weights from the reconstruction model to the classification model. UMAP visualizations further reveal discrepancies in the representations of native and weight-transferred classification models [8].

2 Model Architectures

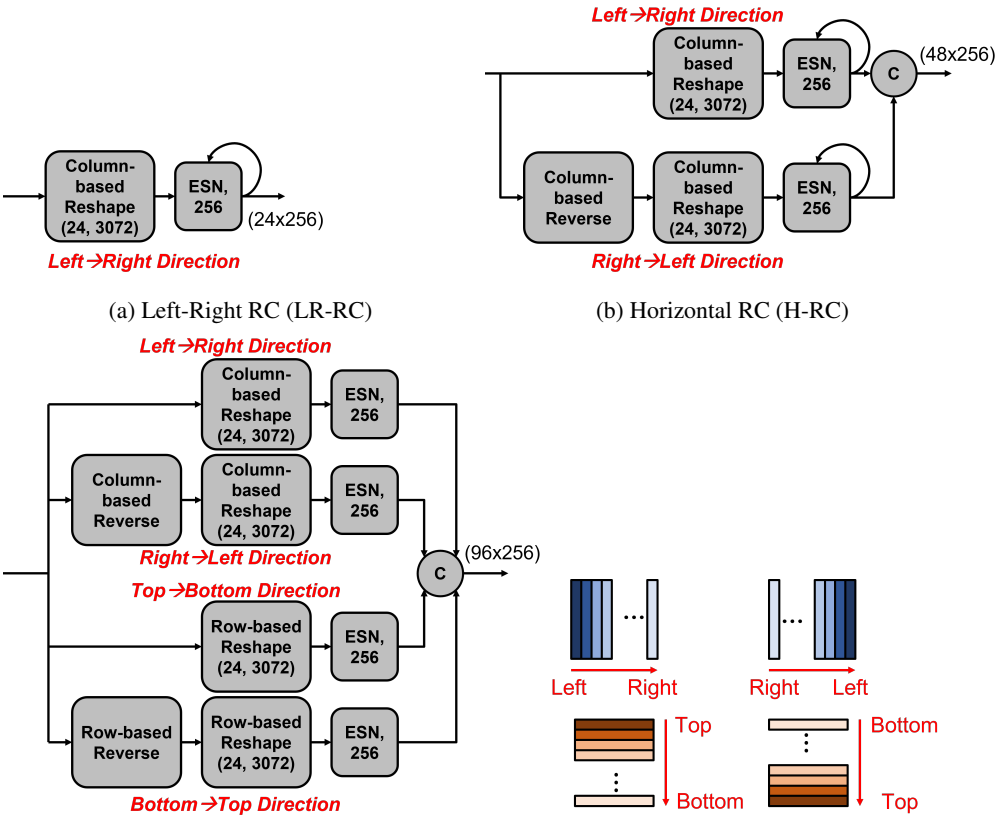
Deep learning models for computer vision typically rely on convolutional layers due to their ability to extract local features, which are crucial for understanding complex visual representations. Meanwhile, reservoir computing has shown promise in capturing sequential dependencies in time series data. By combining convolutional layers with reservoir computing, we aim to leverage the strengths of both approaches—extracting spatial features while maintaining contextual relationships across different regions of the image [9]. Figure 1 depicts the network architecture of our classification and reconstruction model.

2.1 Classification Model

The classification model extracts spatial and sequential features using convolutional layers and reservoir computing. It consists of three convolutional blocks with batch normalization and ReLU activation, where the first block uses a 5×5 kernel with a stride of 2, and subsequent blocks use 3×3 kernels with max-pooling applied in the final block for dimensionality reduction. The number of output channels, controlled by the parameter $F_{conv} = 32$, doubles after each block. The reservoir computing (RC) block first reshapes the feature maps into "spatial sequences" and processes them in a recurrent manner. As mentioned earlier, images do not have an evolutionary direction such as time series. Therefore a straightforward application of ESN to these "spatial sequences" can only keep tracks of contextual dependencies along a single 1D direction, ignoring 2D spatially contextual information. To compensate this lack of context features, we propose a specific bidirectional RC block depicted in Figure 2. Concretely, Figure 2a shows the baseline Left$\longrightarrow$Right RC configuration which first reshapes the 3D input tensor of shape $24 \times 24 \times 128$ into a "sequence" of columns of shape 24×3072. Then it processes this sequence in a recurrent manner to produce a 2D tensor of shape 24×256. Figure 2b depicts the horizontal RC which also processes the "sequence" in the reverse Right$\longrightarrow$Left direction to provide a fully horizontal context. In the same vein, Figure 2c additionally provides vertical context with Top$\longrightarrow$Bottom and Bottom$\longrightarrow$Top directions. The resulting features of these directional ESN processing is concatenated together before being linearly combined together via a fully-connected layer (see Figure 1) to provide the embedding tensor. Figure 2d demonstrate the way the input tensor is reshaped and processed along those aforementioned directions. In details, the number of trainable parameters is only $100k$ for all three configurations.

2.2 Reconstruction Model

The inpainting model follows an encoder–decoder architecture, in which the encoder has the same design as the one in the classification model and incorporates the S-RC block to obtain robust contextual feature processing. These features are then passed to the decoder, which consists of fully connected layers and a multi-head attention (Att) to capture long-range dependencies between regions. The features are then reshaped to 3D tensor and passed through several convolutions and pixel shuffles to gradually increase the spatial dimension up to the original dimension. A mask prediction branch generates a binary mask to identify corrupted regions, ensuring only missing parts are reconstructed while preserving unaltered areas. The final output combines the reconstruction with



(a) Left-Right RC (LR-RC)

(b) Horizontal RC (H-RC)

(c) Spatial RC (S-RC)

(d) Evolutionary directions

Figure 2: Proposed RC blocks with different evolutionary directions.

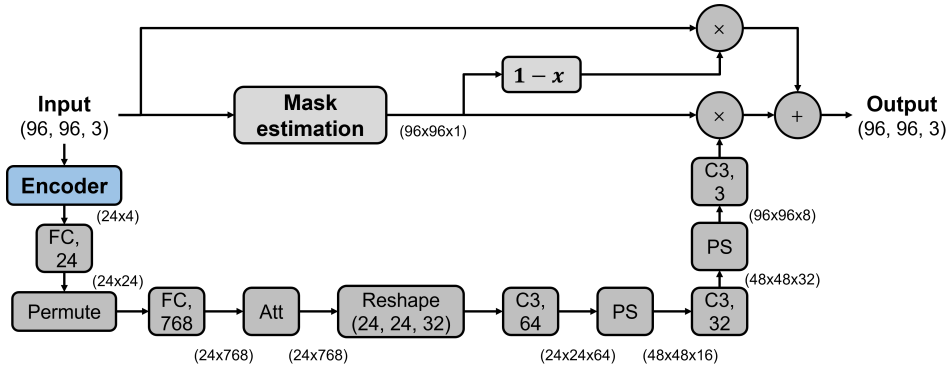


Figure 3: Architecture of the reconstruction model.

the original input, using the mask to retain non-corrupted pixels. In details, the entire reconstruction model containts nearly $460k$ trainable parameters.

3 Transfer Learning Strategy

Existing works have demonstrated the effectiveness of transfer learning [10, 11]. For the sake of efficiency, in this project we also investigate the possibility of transferring the encoder learned for one task to another task. The approach involves matching layer names between the source (trained) model and the target model, and then setting the target model's weights accordingly. This strategy was applied in two directions:

- C2I: To evaluate if features learned for Classification can be used for image Inpainting.
- I2C: To assess whether embedding representations from the Inpainting task are useful for the Classification task.

4 Experiments

4.1 Dataset

The STL10 dataset comprises color images with a resolution of 96×96 pixels and is divided into training and testing sets. For this project, both classification and reconstruction tasks use the same underlying dataset [12]. For the classification task, pixel values in images are normalized to the [0, 1] range. Data augmentation techniques—such as padding, random cropping, horizontal flipping, and cutout—are applied to increase the diversity of training examples and improve generalization. In the reconstruction task, images are similarly normalized. A corruption process is applied by generating a random pixel mask. For the training dataset, the corruption ratio is randomly chosen between 0.25 and 0.75 to simulate missing regions. For the evaluation dataset, the corruption ratio is set to 0.25, 0.5, or 0.75, or randomly selected within the same range. This prepares the data for an inpainting model that must reconstruct the original image from its corrupted version [13].

4.2 Training Configurations

Each model was trained for 100 epochs using the Adam optimizer with categorical cross-entropy for classification and mean absolute error (MAE) for reconstruction. The initial learning rate was set to 10^{-3} and reduced using a cosine decay scheduler. Early stopping was applied with a patience of 20 epochs, restoring the model with the best loss. After initial training, weight transfer experiments were conducted, and the transferred models were further fine-tuned for 100 epochs.

4.3 Evaluation Metrics

The evaluation metrics differed based on the task:

- Classification: Accuracy and UMAP visualization for feature exploration [8].
- Reconstruction: Peak Signal-to-Noise Ratio (PSNR), and Structural Similarity Index (SSIM).

4.4 Results

4.4.1 Classification Results

Table 1 presents the experimental result of the classification model. Three RC configurations were evaluated, with S-RC indeed providing the best validation performance. As a result, this bidirectional RC architecture was used for all subsequent experiments. When weights were transferred from the reconstruction model to the classification model (I2C), a significant performance drop was observed compared to the originally trained classification model. This can be explained by the fact that the encoder only learns low-level feature for missing pixels in the inpainting task, and these features are not directly suitable for classification tasks and require more flexible fine-tuning to improve the performance.

UMAP was used to visualize the feature embeddings of the classification models in order to compare the native classification model with the weight-transferred classification model (Figure 4):

- *Native Classification Model*: The UMAP projection shows well-separated clusters, with each class forming distinct and compact groups. This clear separation indicates that the native model's embedding space is highly discriminative, effectively capturing class-specific features. The organized latent space aligns with the relatively high validation accuracy ($\sim 69.76\%$) observed in the experiments.
- *Weight-Transferred (I2C) Classification Model*: In contrast, the UMAP visualization of the weight-transferred model reveals significant overlap among clusters. The boundaries between classes are blurred, suggesting that the feature representations are less distinct.

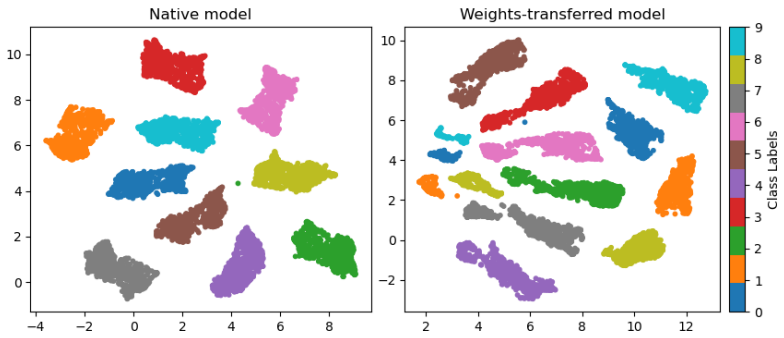


Figure 4: UMAP visualization of data classification and its transferred counterpart.

This intermingling of classes corresponds with the marked drop in accuracy ($\sim 48.40\%$) observed in the quantitative results, indicating that the transferred weights do not preserve the class-specific discriminative power.

Table 1: Performance of classification model.

Model config.	**Acc (%)**
LR-RC	67.95
H-RC	68.20
S-RC	69.76
S-RC (I2C)	48.40

4.4.2 Reconstruction Results

Result of the reconstruction model using Config. 3 are depicted in Figure 5. In general, the native model demonstrated a clear performance trend across different corruption ratios (Table 2). At a low corruption ratio of 0.25, it achieved the highest reconstruction quality with a PSNR of 28.58 and an SSIM of 90.60%. As the corruption ratio increased, the PSNR dropped to 24.98 and the SSIM to 80.63% at 0.50, and at the highest corruption ratio of 0.75, the model struggled to recover details, reaching a PSNR of 20.53 and an SSIM of 59.25%. When evaluated with a randomly chosen corruption ratio within the range [0.25-0.75] for each image, the native model achieved an average PSNR of 24.85 and an SSIM of 78.84%.

In comparison, the weight-transferred model consistently underperformed across all corruption levels. At 0.25 corruption, it obtained a PSNR of 27.56 and an SSIM of 87.50%, showing a slight degradation from the native model. This gap widened as corruption increased, with the transferred model achieving 24.16 PSNR and 77.27% SSIM at 0.50 corruption, and 19.01 PSNR and 53.75% SSIM at 0.75 corruption. Over the full range [0.25-0.75], its average PSNR was 23.88 and SSIM 75.13%, confirming its weaker performance.

These results indicate that the native model retains superior reconstruction ability, likely due to better adaptation to the reconstruction task. In contrast, the weight-transferred model struggles to generalize, highlighting the need for fine-tuning when transferring features from classification to reconstruction.

5 Conclusion

This project presents a comprehensive study on the use of convolutional-reservoir architectures for image classification and reconstruction on the STL10 dataset, along with an investigation into weight transfer between these tasks. The key findings are as follows:

- *Classification*: Among the three configurations tested, S-RC achieved the best validation accuracy ($\sim 69.76\%$) with a gain of nearly 2% compared to the baseline LR-RC architecture. This demonstrates the robustness of the our bidirectional computation structures to encode spatially contextual dependencies using ESN.

Table 2: Performance of reconstruction model.

Model config.	**Corruption ratio**	**PSNR**	**SSIM (%)**
Config. 3, native	0.25	28.57	90.58
$(3.8M)$	0.50	24.98	80.63
	0.75	20.53	59.25
	[0.25-0.75]	24.85	78.84
Config. 3, transferred	0.25	27.56	87.49
$(3.8M)$	0.50	24.16	77.27
	0.75	19.01	53.76
	[0.25-0.75]	23.88	75.13

- *Reconstruction*: The native reconstruction model delivered high-quality image recovery with strong PSNR and SSIM metrics. However, when weights were transferred from the classification model, performance slightly declined.
- *Weight Transfer*: Directly transferring weights from one task to another—particularly from reconstruction to classification—resulted in a notable drop in accuracy. UMAP visualizations further confirmed that the weight-transferred model's latent space is less discriminative compared to the native model.

Future Work: Future research should explore advanced transfer learning techniques or joint training approaches that better align the learned representations across tasks, potentially incorporating fine-tuning steps to adapt transferred weights more effectively.

References

[1] J.-L. Faulon, P. Ahavi, and T.-N.-A. Hoang, "Reservoir computing with bacteria," *bioRxiv*, 2024. [Online]. Available: https://www.biorxiv.org/content/early/2024/09/12/2024.09.12.612674

[2] V. Nikolić, M. Echlin, B. Aguilar, and I. Shmulevich, "Computational capabilities of a multicellular reservoir computing system," *PLOS ONE*, vol. 18, no. 4, pp. 1–19, 04 2023. [Online]. Available: https://doi.org/10.1371/journal.pone.0282122

[3] P. Baldini, "Reservoir computing in robotics: a review," 2022. [Online]. Available: https://arxiv.org/abs/2206.11222

[4] R. Budhiraja, M. Kumar, M. K. Das, A. S. Bafila, and S. Singh, "A reservoir computing approach for forecasting and regenerating both dynamical and time-delay controlled financial system behavior," *PLOS ONE*, vol. 16, no. 2, pp. 1–24, 02 2021. [Online]. Available: https://doi.org/10.1371/journal.pone.0246737

[5] E. Picco, A. Lupo, and S. Massar, "Deep photonic reservoir computer for speech recognition," *IEEE Transactions on Neural Networks and Learning Systems*, p. 1–9, 2024. [Online]. Available: http://dx.doi.org/10.1109/TNNLS.2024.3400451

[6] E. J. López-Ortiz, M. Perea-Trigo, L. M. Soria-Morillo, F. Sancho-Caparrini, and J. J. Vegas-Olmos, "Exploring deep echo state networks for image classification: a multi-reservoir approach," *Neural Computing and Applications*, vol. 36, no. 20, pp. 11 901–11 918, Jul. 2024. [Online]. Available: https://doi.org/10.1007/s00521-024-09656-4

[7] Z. Tong and G. Tanaka, "Reservoir Computing with Untrained Convolutional Neural Networks for Image Recognition," in *2018 24th International Conference on Pattern Recognition (ICPR)*, Aug. 2018, pp. 1289–1294, iSSN: 1051-4651. [Online]. Available: https://ieeexplore.ieee.org/document/8545471

[8] L. McInnes, J. Healy, and J. Melville, "Umap: Uniform manifold approximation and projection for dimension reduction," 2020. [Online]. Available: https://arxiv.org/abs/1802.03426

[9] G. Tanaka, T. Yamane, J. B. Héroux, R. Nakane, N. Kanazawa, S. Takeda, H. Numata, D. Nakano, and A. Hirose, "Recent advances in physical reservoir computing: A review," *Neural Networks*, vol. 115, pp. 100–123, 2019. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0893608019300784

[10] H. M. Fayek, L. Cavedon, and H. R. Wu, "On the transferability of representations in neural networks between datasets and tasks," 2018. [Online]. Available: https://arxiv.org/abs/1811.12273

[11] J. Yosinski, J. Clune, Y. Bengio, and H. Lipson, "How transferable are features in deep neural networks?" in *Advances in Neural Information Processing Systems*, Z. Ghahramani, M. Welling, C. Cortes, N. Lawrence, and K. Weinberger, Eds., vol. 27. Curran Associates, Inc., 2014. [Online]. Available: https://proceedings.neurips.cc/paper_files/paper/2014/file/532a2f85b6977104bc93f8580abbb330-Paper.pdf

[12] A. Coates, A. Ng, and H. Lee, "An analysis of single-layer networks in unsupervised feature learning," in *Proceedings of the Fourteenth International Conference on Artificial Intelligence and Statistics*, ser. Proceedings of Machine Learning Research, G. Gordon, D. Dunson, and M. Dudík, Eds., vol. 15. Fort Lauderdale, FL, USA: PMLR, 11–13 Apr 2011, pp. 215–223. [Online]. Available: https://proceedings.mlr.press/v15/coates11a.html

[13] D. Pathak, P. Krähenbühl, J. Donahue, T. Darrell, and A. A. Efros, "Context encoders: Feature learning by inpainting," in *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016, pp. 2536–2544.

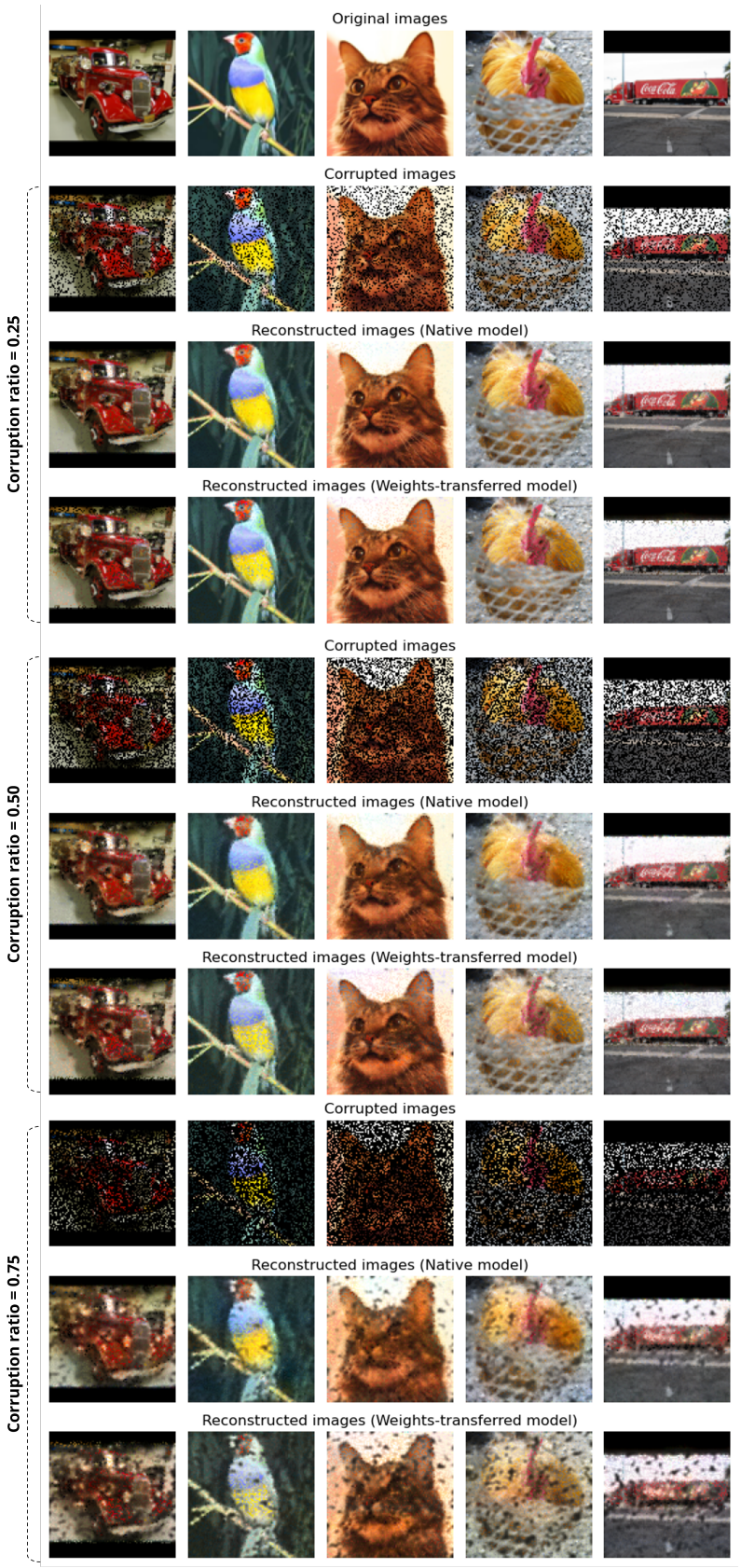


Figure 5: Illustration of reconstruction model result.