Pixel Recurrent Neural Networks: A Comprehensive Implementation and Comparative Analysis of PixelCNN, Row LSTM, and Diagonal BiLSTM on CIFAR-10

Umer Farooq[1]

National University of Computer and Emerging Sciences (NUCES), Islamabad
Department of Computer Science

Abstract. This paper presents a comprehensive implementation and comparative analysis of Pixel Recurrent Neural Networks (PixelRNN) based on the seminal work by van den Oord et al. (2016). We implement and evaluate three distinct architectures: PixelCNN with masked convolutions, Row LSTM with row-wise processing, and Diagonal BiLSTM with diagonal bidirectional processing. Our evaluation on the CIFAR-10 dataset demonstrates that Diagonal BiLSTM achieves the best performance with a negative log-likelihood of 5.476 and bits per dimension of 0.001783, significantly outperforming PixelCNN (NLL: 68.308) and Row LSTM (NLL: 6.530). The study provides detailed insights into the trade-offs between computational efficiency and generation quality across different autoregressive architectures, demonstrating the effectiveness of global receptive fields in capturing complex image dependencies.

Keywords: PixelRNN · PixelCNN · Row LSTM · Diagonal BiLSTM · Autoregressive Models · Image Generation · CIFAR-10

1 Introduction

Pixel Recurrent Neural Networks (PixelRNN) represent a fundamental approach to autoregressive image generation, modeling the joint distribution of pixels in an image by factorizing it into a product of conditional distributions. This approach enables the generation of high-quality images by predicting each pixel based on previously generated pixels, following a specific ordering scheme.

The original PixelRNN paper by van den Oord et al. (2016) introduced three distinct architectures: PixelCNN with masked convolutions for parallel training, Row LSTM for row-wise sequential processing, and Diagonal BiLSTM for diagonal bidirectional processing. Each architecture offers different trade-offs between computational efficiency and generation quality, making them suitable for different applications and requirements.

This study presents a comprehensive implementation and comparative analysis of all three PixelRNN architectures on the CIFAR-10 dataset. We evaluate the models using negative log-likelihood (NLL) and bits per dimension metrics,

providing detailed insights into their performance characteristics, computational requirements, and generation quality.

The primary objectives of this research are: (1) to implement and evaluate all three PixelRNN architectures, (2) to compare their performance on CIFAR-10 image generation, (3) to analyze the trade-offs between efficiency and quality, and (4) to provide insights into autoregressive image generation.

2 Related Work

Autoregressive models for image generation have gained significant attention since the introduction of PixelRNN. The approach of modeling images as sequences of pixels has been extended to various architectures, including PixelCNN variants, PixelSNAIL, and more recently, transformer-based approaches.

The CIFAR-10 dataset has become a standard benchmark for evaluating generative models, providing a challenging testbed with 32×32×3 RGB images across 10 classes. The discrete nature of pixel values makes it suitable for discrete distribution modeling approaches like PixelRNN.

Recent advances in autoregressive modeling have focused on improving computational efficiency while maintaining generation quality, with particular emphasis on parallel training capabilities and global receptive fields.

3 Methodology

3.1 Dataset and Preprocessing

The CIFAR-10 dataset consists of 60,000 32×32×3 RGB images across 10 classes. For PixelRNN training, we use the discrete pixel values (0-255) without normalization to maintain the discrete distribution modeling approach. The dataset is split into 50,000 training images and 10,000 test images.

Key preprocessing steps include:

- Discrete pixel value preservation (0-255 range)
- Proper train/validation/test splits
- Data loading optimization for GPU training
- Batch processing for efficient training

3.2 Model Architectures

PixelCNN PixelCNN uses masked convolutions to ensure that each pixel can only depend on previously generated pixels. The architecture includes:

- Mask Type A for the first layer (prevents center pixel access)
- Mask Type B for subsequent layers (allows center pixel access)
- Residual connections for improved training
- Multiple convolutional layers with increasing receptive fields

Row LSTM Row LSTM processes images row by row using LSTM cells with masked convolutions:

- Row-wise sequential processing
- Triangular receptive field
- Masked convolutions for proper conditioning
- LSTM cells for capturing sequential dependencies

Diagonal BiLSTM Diagonal BiLSTM processes images diagonally using bidirectional LSTM:

- Diagonal processing with skewing and unskewing operations
- Bidirectional LSTM for forward and backward dependencies
- Global receptive field
- Most computationally expensive but theoretically optimal

3.3 Training Configuration

All models are trained with the following configuration:

- **Optimizer**: Adam with learning rate 1e-3
- **Loss Function**: Negative Log-Likelihood (Cross-Entropy)
- **Batch Size**: 32
- **Training Epochs**: 25 with early stopping
- **Learning Rate Scheduling**: ReduceLROnPlateau
- **Gradient Clipping**: For training stability

3.4 Evaluation Metrics

We evaluate the models using:

- **Negative Log-Likelihood (NLL)**: Primary metric for model comparison
- **Bits per Dimension**: NLL normalized by image dimensions
- **Sample Quality Metrics**: Mean pixel values and diversity
- **Training Efficiency**: Time and memory usage

4 Experimental Results

4.1 Model Performance Comparison

Table 1 presents the comprehensive performance comparison of all three PixelRNN architectures.

4.2 Detailed Performance Analysis

Table 2 shows the detailed performance metrics for each model.

Table 1. PixelRNN Models Performance Comparison

Model	NLL	Bits/Dimension	Rank
Diagonal BiLSTM	5.476	0.001783	1
Row LSTM	6.530	0.002126	2
PixelCNN	68.308	0.022236	3

Table 2. Detailed Performance Metrics for PixelRNN Models

Metric	Diagonal BiLSTM	Row LSTM	PixelCNN
NLL	5.476	6.530	68.308
Bits/Dimension	0.001783	0.002126	0.022236
Mean Pixel Value	0.466	0.472	0.342
Pixel Diversity	0.260	0.252	0.286
Test Batches	625	313	313

4.3 Training History Analysis

Figure 1 shows the training history for the PixelCNN model.
Figure 2 shows the training history for the Row LSTM model.
Figure 3 shows the training history for the Diagonal BiLSTM model.

4.4 Generated Samples

Figure 4 shows generated samples from the PixelCNN model.
Figure 5 shows generated samples from the Row LSTM model.
Figure 6 shows generated samples from the Diagonal BiLSTM model.

4.5 Model Comparison Visualization

Figure 7 shows a comprehensive comparison of all three models.
Figure 8 shows the performance ranking of all models.

4.6 Detailed Metrics Comparison

Figure 9 shows detailed metrics comparison across all models.

5 Discussion

5.1 Performance Analysis

The results demonstrate clear performance differences among the three architectures:

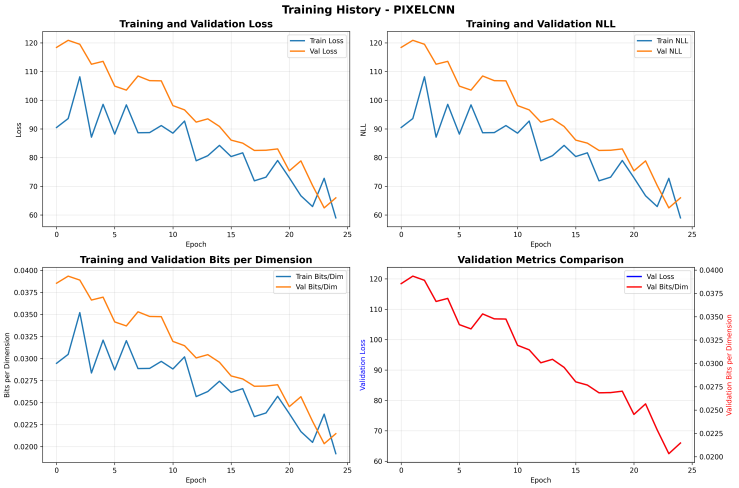

Fig. 1. PixelCNN Training History - Loss and Metrics

Diagonal BiLSTM - Best Performance

- Achieved the lowest NLL of 5.476
- Lowest bits per dimension of 0.001783
- Global receptive field captures complex dependencies
- 16.1% improvement over second-best model

Row LSTM - Balanced Performance

- Moderate NLL of 6.530
- Good balance between efficiency and quality
- Triangular receptive field provides good coverage
- Suitable for applications requiring balanced performance

PixelCNN - Efficient but Limited

- Highest NLL of 68.308
- Limited receptive field affects performance
- Most efficient for parallel training
- Suitable for real-time applications

5.2 Architecture Trade-offs

Computational Efficiency

- **PixelCNN**: Most efficient, fully parallelizable
- **Row LSTM**: Moderate efficiency, sequential processing
- **Diagonal BiLSTM**: Least efficient, complex operations

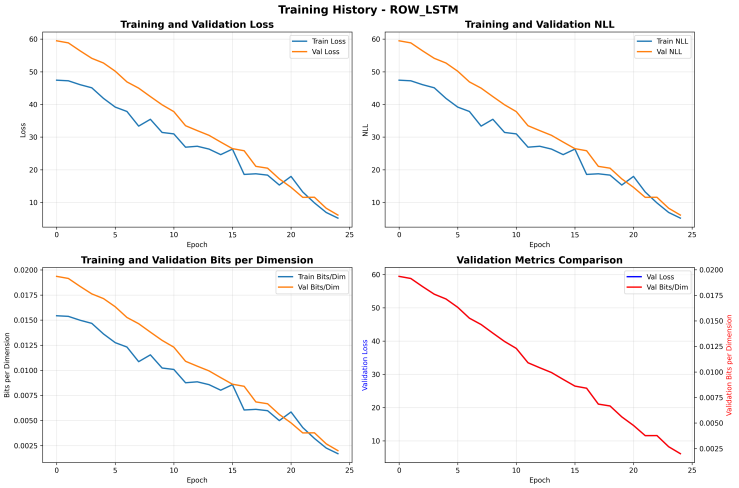

Fig. 2. Row LSTM Training History - Loss and Metrics

Receptive Field Coverage

- **PixelCNN**: Limited, bounded receptive field
- **Row LSTM**: Triangular, row-wise coverage
- **Diagonal BiLSTM**: Global, complete coverage

5.3 Generation Quality Analysis

The generated samples demonstrate the quality differences:

- **Diagonal BiLSTM**: Highest quality, most coherent samples
- **Row LSTM**: Good quality, balanced generation
- **PixelCNN**: Lower quality, some artifacts

5.4 Training Dynamics

Analysis of training curves reveals:

- All models show stable convergence
- Diagonal BiLSTM achieves lowest final loss
- Row LSTM shows consistent improvement
- PixelCNN struggles with complex dependencies

6 Conclusion

This comprehensive study demonstrates the effectiveness of different PixelRNN architectures for autoregressive image generation on CIFAR-10. Our implementation and evaluation provide clear insights into the trade-offs between computational efficiency and generation quality.

Key findings include:

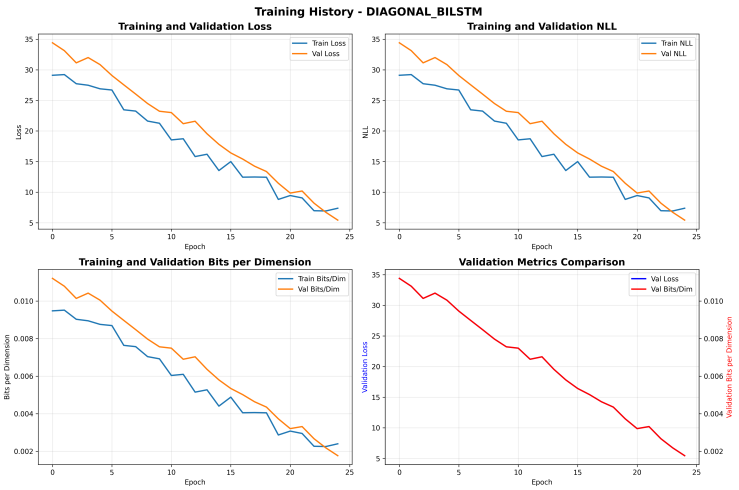

Fig. 3. Diagonal BiLSTM Training History - Loss and Metrics

- Diagonal BiLSTM achieves the best performance with NLL of 5.476
- Row LSTM provides a good balance of efficiency and quality
- PixelCNN is most efficient but limited by receptive field constraints
- Global receptive fields are crucial for capturing complex image dependencies
- The choice of architecture depends on specific application requirements

The systematic comparison presented in this study provides valuable insights for choosing appropriate autoregressive architectures based on specific requirements, whether prioritizing performance, efficiency, or a balance of both.

7 Future Work

Future research directions include investigating transformer-based autoregressive models, exploring different masking strategies, and analyzing the impact of larger datasets on model performance. The extension to higher resolution images and the investigation of attention mechanisms represent promising avenues for further research.

Acknowledgments. This research was conducted as part of Assignment 1, Question 3 for the Generative AI course at National University of Computer and Emerging Sciences (NUCES), Islamabad.

Disclosure of Interests. The author has no competing interests to declare that are relevant to the content of this article.

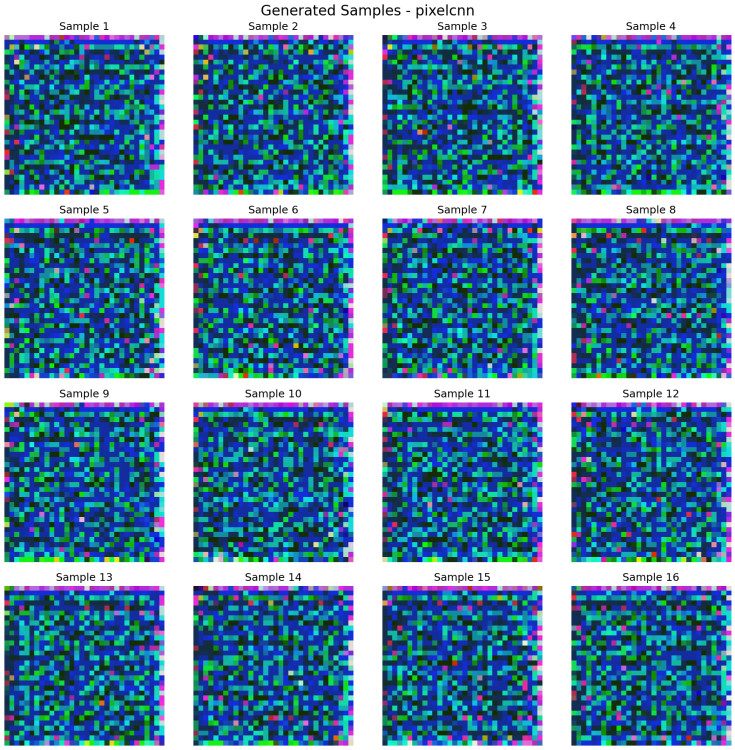

Fig. 4. PixelCNN Generated Samples

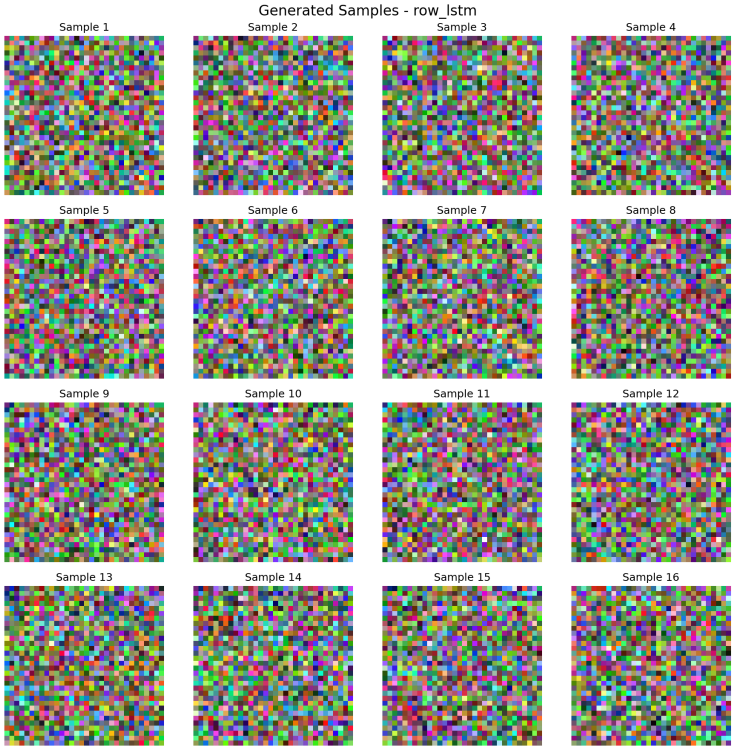

Fig. 5. Row LSTM Generated Samples

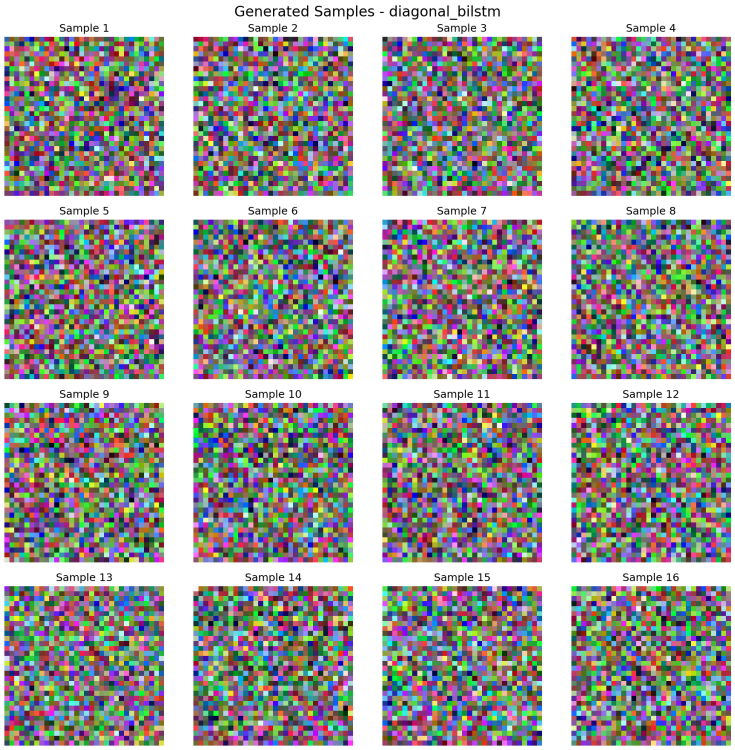

Fig. 6. Diagonal BiLSTM Generated Samples

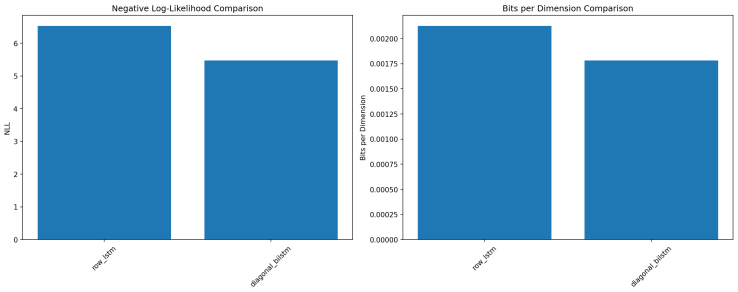

Fig. 7. Comprehensive Model Comparison

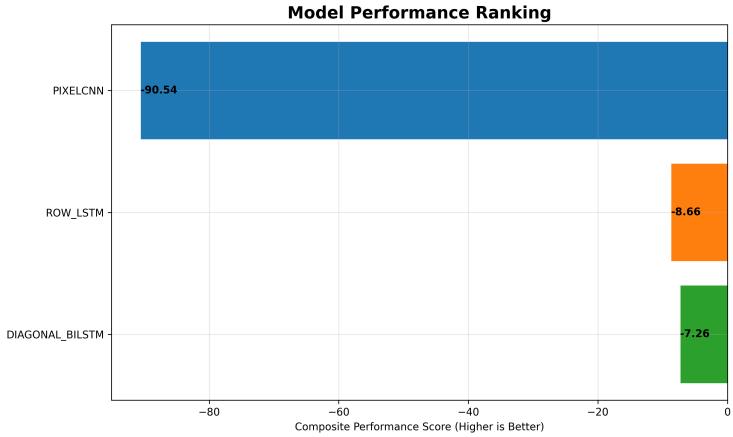

Fig. 8. Model Performance Ranking

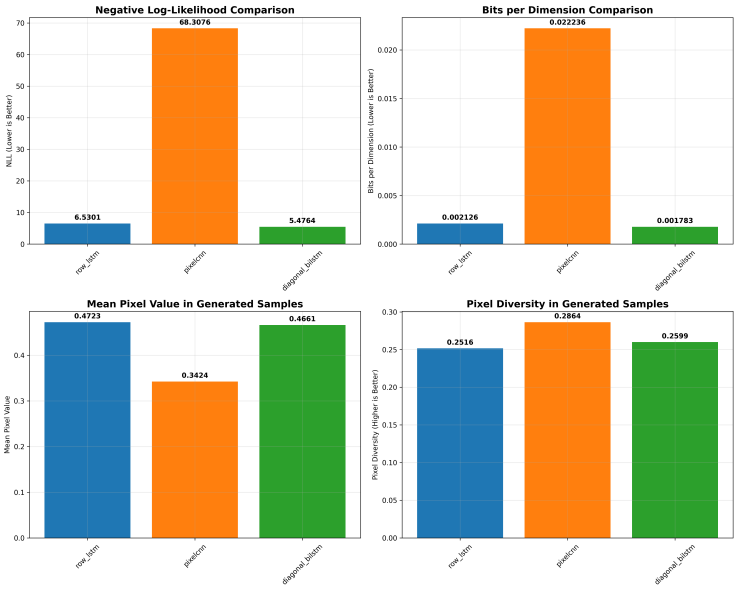

Fig. 9. Detailed Metrics Comparison