Addressing Implementation Challenges in Deep Learning Video Prediction Models

Ibrahim Akhtar
i211679
Solo Memeber
Did it alone
That's why it is bad

***Abstract*—This paper (I say paper but I mean report everywhere) presents the implementation challenges and solutions encountered while developing three video prediction models: Enhanced ConvLSTM with temporal attention, PredRNN with spatiotemporal memory, and a Recurrent Memory Transformer. We address critical issues including model overfitting, dimensional consistency, training efficiency, and temporal coherence. Our solutions reduced training time from 87 to 12-15 hours while maintaining model performance. The paper details specific technical approaches to memory management, convergence optimization, and training stability. I developed these three distinct approaches: an Enhanced ConvLSTM with temporal attention inspired by [1], a PredRNN with spatiotemporal memory [2], and a Recurrent Memory Transformer based on [3].**

***Index Terms*—video prediction, deep learning, ConvLSTM, PredRNN, transformer, optimization**

I. Introduction

Video frame prediction presents unique challenges in deep learning implementations. I developed three distinct approaches: an Enhanced ConvLSTM with temporal attention, a PredRNN with spatiotemporal memory, and a Recurrent Memory Transformer. Each model encountered specific challenges requiring innovative solutions.

II. Implementation Challenges and Solutions

A. Model Overfitting

Initial implementations showed significant overfitting, with validation loss diverging after approximately 15 epochs. I addressed this through multiple techniques:

- Implemented dropout layers (0.3) in ConvLSTM cells
- Added data augmentation with random horizontal flips, color jittering, and rotations
- Incorporated L2 regularization with weight decay in optimizers
- Implemented early stopping with a patience of 10 epochs

B. Dimension and Channel Consistency

A persistent challenge was maintaining consistent dimensions and channel configurations across models, particularly in the PredRNN implementation. I implemented the following solutions:

- Created channel validation checks through explicit verification functions
- Standardized input processing across all models
- Developed uniform channel conversion utilities

C. Training Efficiency

Initial training iterations required approximately 87 hours. I optimized this to 12-15 hours through:

- Implementation of gradient accumulation with 4-step accumulation
- Mixed precision training using torch.cuda.amp
- Optimization of batch sizes and worker counts based on available GPU memory

D. Memory Management

The Transformer model particularly suffered from memory constraints with longer sequences. Solutions included:

- Implementation of gradient checkpointing for memory-efficient backpropagation
- Systematic memory clearing between training iterations
- Optimization of caching strategies for intermediate computations

E. Convergence Optimization

To address slow convergence and local minima issues, I implemented:

- Cyclical learning rates with base_lr=1e-4 and max_lr=1e-3
- 5-epoch warm-up period with gradual learning rate scaling
- Gradient clipping with max_norm=1.0

F. Temporal Coherence

To improve the temporal consistency of predictions, I developed:

- Custom temporal consistency loss function
- Combined MSE and temporal difference penalties
- Weighted loss combinations for balanced optimization

G. Resource Management

To handle memory constraints with large datasets, I developed:

- Lazy loading dataset implementation
- Efficient frame indexing system
- Progressive batch size scheduling

H. Training Stability

For maintaining stable training with larger batch sizes, I implemented:

- Normalized gradient clipping
- Dynamic batch size adjustment
- Systematic learning rate scheduling

III. Quantitative Results

I evaluated the models using Mean Squared Error (MSE) and Structural Similarity Index Measure (SSIM) on different action sequences. Tables I and II present the quantitative results for typing and yo-yo actions respectively.

TABLE I
Performance Metrics for Typing Action Sequence

Model	**MSE**	**SSIM**
Enhanced ConvLSTM	0.099558	0.296273
PredRNN	0.019115	0.785814
Recurrent Memory Transformer	0.034680	0.354711

TABLE II
Performance Metrics for YoYo Action Sequence

Model	**MSE**	**SSIM**
Enhanced ConvLSTM	0.020076	0.372000
PredRNN	0.040034	0.758295
Recurrent Memory Transformer	0.022363	0.456513

The results demonstrate varying performance across different action sequences. For typing actions, the PredRNN model [2] achieved the best performance with an MSE of 0.019115 and SSIM of 0.785814, significantly outperforming other approaches. In the yo-yo sequence, while PredRNN maintained the highest SSIM score of 0.758295, the Enhanced ConvLSTM showed the lowest MSE at 0.020076, indicating better pixel-level accuracy. The Recurrent Memory Transformer [3] maintained consistent mid-range performance across both sequences, suggesting robust generalization capabilities.

These metrics indicate that model performance is highly dependent on the type of motion being predicted, with PredRNN showing particular strength in structural preservation as indicated by the consistently higher SSIM scores. The Enhanced ConvLSTM's variable performance between sequences suggests a sensitivity to the type of motion being predicted.

IV. Results and Discussion

The implemented solutions significantly improved model performance and training efficiency. Training time reduced by approximately 85% while maintaining model accuracy. While some degree of overfitting remains a challenge in all implementations, the combination of techniques described above made the models practically viable for research and development purposes.

V. Conclusion

Through systematic identification and resolution of implementation challenges, I successfully developed three functional video prediction models. The solutions presented demonstrate effective approaches to common deep learning implementation issues, particularly in the context of video prediction tasks.

References

[1] J. Su, W. Byeon, J. Kossaifi, F. Huang, J. Kautz, and A. Anandkumar, "Convolutional Tensor-Train LSTM for Spatio-Temporal Learning," in Advances in Neural Information Processing Systems, vol. 33, pp. 13714–13726, 2020.

[2] Y. Wang, M. Long, J. Wang, Z. Gao, and P. S. Yu, "PredRNN: Recurrent Neural Networks for Predictive Learning using Spatiotemporal LSTMs," in Advances in Neural Information Processing Systems, vol. 30, pp. 879–888, 2017.

[3] A. Bulatov, Y. Kuratov, and M. S. Burtsev, "Recurrent Memory Transformer," arXiv preprint arXiv:2207.06881, 2022.