Short Term Spatiotemporal Video Prediction on Sports via Convolutional LSTMs

Aidan Curtis
Department of Electrical Engineering
Department of Computer Science
Rice University
Houston, TX 77005
arc11@rice.edu

Victor A. Gonzalez
Department of Electrical Engineering
Department of Computer Science
Rice University
Houston, TX 77005
victor.gonzalez@rice.edu

Abstract

Predicting short term video dynamics has many useful applications in self-driving cars, weather nowcasting, and model-based reinforcement learning. Although many video prediction models have been developed and optimized for the moving MNIST dataset, there has been very little exploration of how these models perform on more interesting video prediction tasks. We aim to provide an in-depth analysis of the available models for video prediction and their strengths and weaknesses in predicting natural sequences of images. Specifically, we will be utilizing convolutional long short-term memory (LSTM) cells in various recurrent neural network (RNN) architectures for predicting a sequence of frames with three different sports. We also test strategies for fixing the weaknesses inherent in these existing models. Code Repo: https://github.com/southpawac/PredictiveNetworks

1 Introduction

Video prediction requires the satisfaction of two simultaneous goals. First, the model must maintain spatial coherence by understanding the underlying substructures within the video second, the model must be able to make use of historical evidence to predict temporal dynamics of those assumed substructures. The first large step in video prediction was the creation of ConvLSTM [1] which attempted to create an action conditioned video prediction model by creating transformations on the original image with mask compositions. This was the first introduction to the idea of a convolutional LSTM cell. Following the result found in this paper we decided the first model to test was a shallow network made of a single layer of ConvLSTMs. The next key breakthrough was found by adding several layers of these ConvLSTMs to create a "stacked" architecture. This added depth improved the network's capability for spatial complexity [2, 3]. Another came via PredRNN, which modified the ConvLSTM to create a Spatiotemporal LSTM to fully integrate temporal and spatial memory, allowing information belonging to different layers of the stacked LSTM to interact [4]. PredRNN++ further improved upon PredRNN by adding a "gradient highway" which fixed the gradient back-propagation issue and introduced a Causal-LSTM which adds depth to the recurrence between time points [5].

While these papers all demonstrated advancements in predicting spatiotemporal motion, their architectures have had limited application to the "moving MNIST" dataset, which consists of multiple numbers moving around a 64x64 image. However, this dataset does not provide sufficient complexity to qualitatively analyze the advantages and shortcomings of various future prediction models.

The use case that this paper focuses on is three different sports: tennis, volleyball and soccer. We will attempt to apply the aforementioned architectures to the different sports to determine how close modern attempts can come to correct predictions. Our ultimate goal is to demonstrate that it is possible for a neural network to predict future frames of a sports game, which contains a considerable

amount of noise and variance. We also aim to come up with a better loss function for training which maximizes perceptual similarity to the ground truth.

2 Methods

2.1 Data Preprocessing

In order to limit the data to important gametime, we went through the games for each sport and spliced out any time where there was no motion or the game was not in play. We then downsampled the video to 160x88 using ffmpeg and converted each video to a uniform 10fps. Each training/testing input required 20 frames. The first 10 frames of the video (one second) was fed into the model as input. Then each of the tested models would predict the next 10 frames of the video. Using L2 loss as a metric of similarity, we compared the predicted 10 frames to the ground truth 10 frames and then backpropagated the error into the network.

We trained 4 different architectures to convergence and performed a qualitative and quantitative analysis on each model's performance on 3 different sports. The first sport we chose was tennis. We chose tennis as the first data distribution because the entire game is recorded from a stationary camera and there is relatively little motion. To get high accuracy, any model only needs to copy the background and understand the trajectory of the player in the foreground. The second sport we chose was volleyball. This is more complicated than tennis because there are many players that move around the screen. Finally, our most complex task was future prediction on soccer. This is clearly the most complex task because the camera is not fixed so the model can no longer rely on a fixed background and must understand camera angle as well as track many players simultaneously.

2.2 ConvLSTM

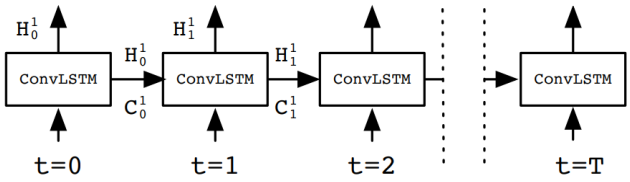

Figure 1: ConvLSTM RNN Architecture [5]

The first architecture we used was a convolutional LSTM RNN. We can see the general structure in Figure 1 is exactly the same as any standard RNN, with the only real difference being the cells used. As stated in [2], by transforming the input into a 3-dimensional tensor on which row and column a point is within, the cell can encode spatial information. We decided this was a good starting architecture to work with, as RNNs perform well at sequence-to-sequence tasks, but the new cell should perform better on images than a simple fully-connected cell.

This approach's largest advantage in comparison to the later two models we experiment with is it's simplicity. Though the capacity is not as large, it trains very quickly due to the smaller nature. However, the lack of depth means that some spatiotemporal data is not being considered.

2.3 Stacked ConvLSTM

In order to combat some of the issues presented with the ConvLSTM architecture, we also wrote a stacked architecture that can be seen in Figure 2. Essentially we take Figure 1 and duplicate the layers, adding cells to the network. By adding more ConvLSTM cells to the network we are enabling deeper spatial dependencies to develop between the input and output image. Now, it is possible to predict finer-grain changes. However, this model begins to suffer from vanishing gradients.

2.4 PredRNN++

PredRNN++ is built to combat the vanishing gradient issue, as well as enhance the actual LSTM cell that is used [5]. We can see the general architecture in Figure 3 is quite similar to Figure 2, but has an

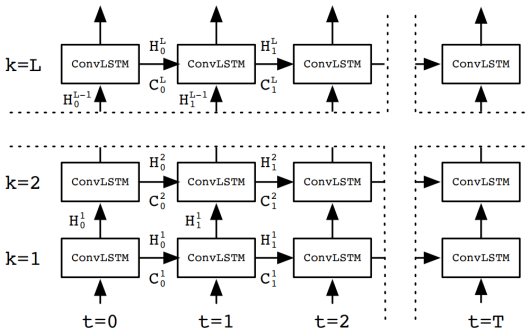

Figure 2: Stacked ConvLSTM RNN Architecture [5]

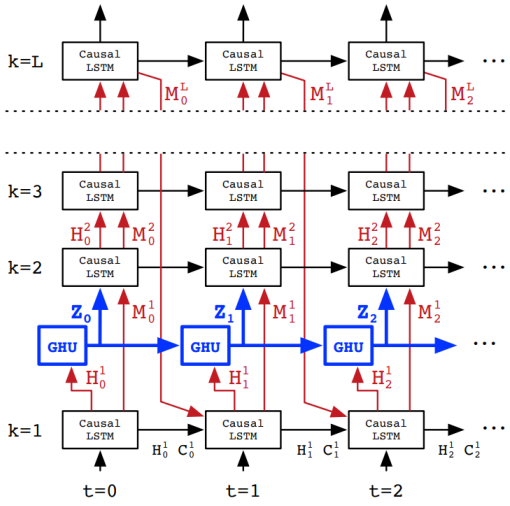

Figure 3: PredRNN++ Architecture [5]

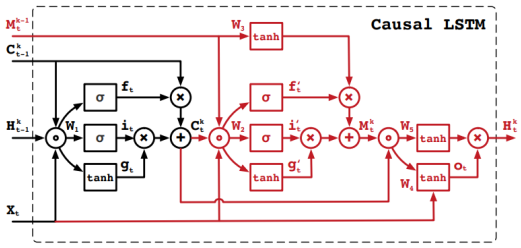

Figure 4: PredRNN++ Architecture [5]

extra state that is passed through M, which is for the Causal LSTM cell defined in Figure 4. The proposed solution to the vanishing gradient problem is a gradient highway, which creates a shorter route for future outputs to backpropagate to distant inputs. The Causal LSTM is an innovation that first appeared in [4], but was duplicated in PredRNN++, which splits the traditional memory into a temporal, C, and spatial, M, memory.

2.5 Different Loss Function

One key area we thought could yield improvement was changing the loss function that all these networks utilize, which is the L2 loss. An issue that usually arises with attempting to utilize that loss function in images is that we suffer from a regression to the mean. From a quantitative standpoint, the loss may be lower with the L2 loss than a different loss, but qualitatively this usually means the image is blurry and details are lost. We decided to try and experiment with using the Structural Similarity (SSIM) between images. SSIM attempts to measure how perceptually similar two images are by taking windows of the image and measuring luminance, contrast, and structure within each window.

3 Experiments

3.1 Dataset

Sport Dataset Size		
Sport	Training Set Clips	Validation/Test Set Clips
Tennis	1391	348
Volleyball	1637	410
Soccer	665	167

Table 1: Size of datasets used

For the various sports, we decided to only use data from one game for each sport. The main reason we decided upon this approach was to reduce the number of "parameters" that the various models would need to learn, such as field color, uniform color, or time of day. Instead, we pose a problem that is more challenging than a simple moving MNIST dataset, but not overly so. By using sports footage, we still introduce many challenges such as a more than two moving objects, different colored moving objects and in some cases changing camera angle.

For each game, we extracted 2-second clips, which come out to 20 distinct frames. Table 1 displays the number of clips that we used for training and testing in the various sports.

3.2 Experiment 1: ConvLSTM

For this architecture, the only hyperparameter of note was that the hidden unit size for the cell was 64 units. We trained this model for 80,000 batches on each of the sports, with each batch containing 4 samples randomly selected from the train set. The 3 models trained all were trained using the L2 loss.

After finishing training, the models had a loss of 24921, 29595 and 36402 on tennis, volleyball and soccer, respectively.

3.3 Experiment 2: Stacked ConvLSTM

Moving on to the Stacked ConvLSTM model, our architecture contained 4 layers, with the cell in each layer containing (128,64,64,64) hidden units respectively. The models were trained for 80,000 batches on each of the sports, and each batch contained 4 samples randomly selected from the train set. Once again, the 3 models were trained using the L2 loss.

After finishing training, the models had a loss of 23177, 31607 and 33387 on tennis, volleyball and soccer, respectively.

3.4 Experiment 3: PredRNN++

The final architecture model identical to the Stacked ConvLSTM, as we once again had 4 layers with (128,64,64,64) hidden units respectively. We trained the models for 80,000 batches on each of the sports, and the batch consisted of 4 samples randomly drawn from the training set. The 3 models were trained using the L2 loss.

After finishing training, the models had a loss of 22583, 28326, 36884 on tennis, volleyball and soccer, respectively.

3.5 Experiment 4: SSIM Loss Optimization

In this experiment, we took the model that performed best on the most difficult sport, Stacked ConvLSTM on Soccer and switched out the L2 loss for the SSIM loss. The architecture was entirely the same, with the only difference being the loss function we trained on. We can see in Figure 5 the qualitative improvement of the SSIM loss usage over L2 loss usage in a validation image. We did not obtain the L2 loss figure for the SSIM model as it would obviously be higher than the L2 variant.

Figure 5: Stacked ConvLSTM with L2 and SSIM loss for comparison

4 Results and Observations

When combining all of the results together, we can see in Table 2 which models achieve the lowest L2 loss over the given sport's test set. These numbers help give a quantitative measure to evaluate what predictions are the best.

Model Performance Comparison			
Model	Tennis L2 Loss	Volleyball L2 Loss	Soccer L2 Loss
ConvLSTM	24921 ± 1879	29595 ± 1080	36402 ± 3148
PredRNN++	$\mathbf{22583 \pm 5407}$	$\mathbf{28326 \pm 1934}$	36884 ± 3806
Stacked ConvLSTM	23177 ± 3866	31607 ± 1867	$\mathbf{33387 \pm 3138}$

Table 2: Model performance by mean L2 loss on validation set

The model that ostensibly performs the best on the two fixed-camera sports is PredRNN++, which has the lowest loss by a decent margin in both cases. Stacked ConvLSTM then has the lowest loss on the soccer dataset. We can see some sample outputs from the models in Figures 6, 7, 8. Within each

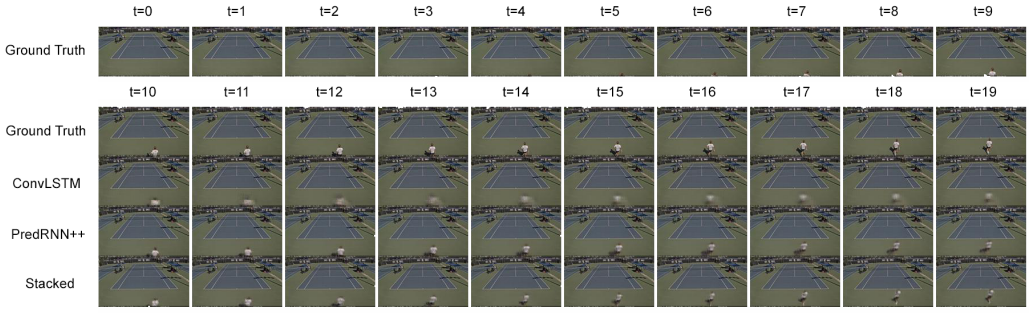

Figure 6: Tennis validation example

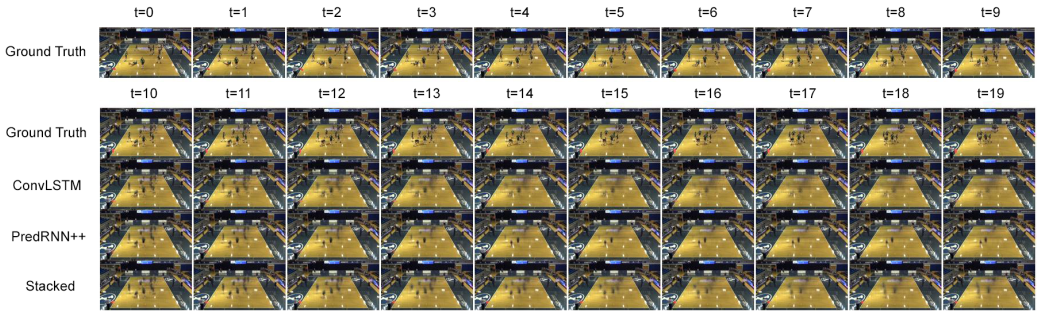

Figure 7: Volleyball validation example

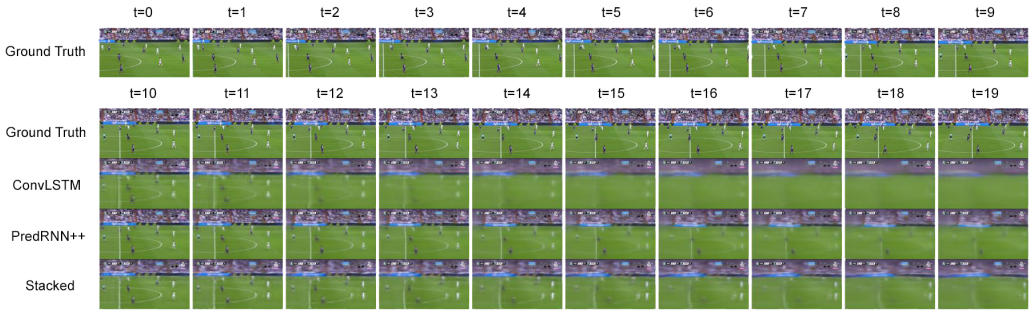

Figure 8: Tennis validation example

figure, the first row shows the ground truth from t=0 to t=9 on a single example. The second row shows the ground truth from t=10 to t=19 on that same example. Rows 3-5 show the predictions of t=10 to t=19 conditioned on the ground truth from t=0 to t=9 for different models, which we label.

Unfortunately, we can easily see where these numbers fall short when examining Figure 8. When checking Tagle 2, we see that PredRNN++ actually has a higher loss than ConvLSTM, which would imply that ConvLSTM produces higher quality images. This is not the case, though, and we can see that when reaching t=17,18,19 the field is just a blurry mess for ConvLSTM, in comparison to PredRNN++ in which you can still see the field markings and some people.

In each of the examples shown in Figures 6, 7, 8, the single layer ConvLSTM is not able to represent any complex visual scenes after decoding. Even at t=10 when the temporal component of prediction is minimized, the single layer model blurs images and cannot fully represent the scene. On the other hand, the stacked ConvLSTM and PredRNN++ are able to fully express the spatial complexity of the input. The most noticeable qualitative difference between the stacked ConvLSTM and PredRNN++ can be seen at t=15-19 in Figure 6. Although both PredRNN++ and stacked ConvLSTM are able to successfully track the motion of the camera and adjust the field position accordingly, Stacked ConvLSTM is not able to account for the nonlinear transformation in time and space that results from the camera angle and motion. While PredRNN++ keeps the line in the center of the field straight, ConvLSTM warps the center line, causing parts of the line closer to the camera to move faster than parts of the line that are further away. This complex interaction between spatial and temporal features of the input is exactly what PredRNN++ was built to account for, even if this difference is not significantly represented in the L2 Loss.

Returning to the issue of blurriness, we are seeing regression to the mean in effect due to the L2 loss used. Often times, the player is reduced to a blob, or in some extreme cases essentially erased. As can be seen in Figure 5, SSIM does a much better job of maintaining perceptual similarity to the ground truth. Because of the frequent shifts in the location of the players and angle/orientation of the camera, optimizing on L2 gives the model a strong incentive to blur the players on the field as t increases and default to the common background of the crowd and green field. On the other hand, SSIM gives a high loss for such predictions because it has low contrast when compared to the ground truth.

5 Conclusion

After examining the various models and experiments, it is clear that PredRNN++ performs the best across the various sports both qualitatively and quantitatively. Moreover, we can see that taking more care to tailor the model to our use case (qualitatively good predictions of frames) by changing the loss to SSIM presents a marked improvement. Especially for tasks like self-driving cars, it is important that we don't have people "disappear" when we predict future frames. However, even the worst model in ConvLSTM is still adequate for the first 3 frames or so.

We successfully trained three different models on a previously untested dataset, and obtained results that implied promise in pursuing these networks further. There were, of course, several notable limitations, the largest of which is that our model only works on one game at the moment. Introducing a variety of games and courts is another challenge that would be interesting to explore in the future. Further, these models can take a decent amount of time to train, which would only increase with a larger dataset.

References

[1] Chelsea Finn, Ian Goodfellow, Sergey Levine
Unsupervised Learning for Physical Interaction through Video Prediction.

[2] Xingjian Shi Zhourong Chen Hao Wang Dit-Yan Yeung
Convolutional LSTM Network: A Machine Learning Approach for Precipitation Nowcasting

[3] Seongchan Kim, Seungkyun Hong, Minsu Joh, Sa-kwang Song
DeepRain: ConvLSTM Network for Precipitation Prediction using Multichannel Radar Data

[4] Yunbo Wang, Mingsheng Long, Jianmin Wang, Zhifeng Gao, Philip S. Yu
PredRNN: Recurrent Neural Networks for Predictive Learning using Spatiotemporal LSTMs

[5] Yunbo Wang, Zhifeng Gao, Mingsheng Long, Jianmin Wang, Philip S. Yu
PredRNN++: Towards A Resolution of the Deep-in-Time Dilemma in Spatiotemporal Predictive Learning