Video Frame Prediction with Deep Learning

Megan Backus, Yiwen Jiang, Dana Murphy
{backusm, yjiang98, dmurphy7} @stanford.edu

Stanford
CS 231n Final Project

Background /Introduction

Predicting future human motion has important applications in many fields, such as photo-to-video conversion and decision-making systems. However, the complex filming factors together with the unpredictability of human intent make this a challenging task for machines.
Previous work using optical flow algorithms and CNN architectures have been proposed and implemented to predict motion of individual pixels, RNN models have been suggested for frame predictions. Recently, both LSTM and GAN based models have achieved impressive results for video frame prediction.
We explore the application of a Convolutional LSTM model that was proposed for the use of precipitation nowcasting and a GAN model on human motion frame prediction and compare the performance of two promising, but very different deep learning techniques in the context of this important problem.

Problem Statement

This project focuses on adapting and evaluating the performances of a Convolutional LSTM network and a GAN model on human motion video frame prediction.

The Convolutional LSTM model has an input of four past frames and predicts one frame, while the GAN model receives six past frames and outputs six future frames.

We visually examine whether the generated frames are a coherent continuation of the input data; quantitatively, we use Mean Standard Error (MSE), Peak Signal-to-Noise Ratio (PSNR) and Structural Similarity (SSIM) to compare the generated frames with the future frames in the original clip.

Dataset

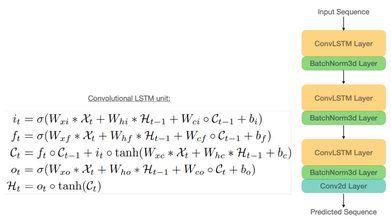
The UCF101 dataset which includes approximately 13,320 video clips corresponding to 101 broad activity classes is used. The dataset is chosen as it provides large variability in both actions and filming features.
For use with the ConvLSTM model, a custom collate function is implemented to remove audio and label data. Pytorch's UCF101 class is used on the resulting video only data to specify desired parameters for the videos including frame rate and steps between clips.
For the FutureGAN model, data is processed with FFmpeg library that split the video into component RGB frames with specific frame rate and image sizes.

Sample processed frame data

Methods

Convolutional LSTM

- Modified LSTM to capture spatial and temporal correlations in video data
- The Convolutional LSTM cell follows the original LSTM cell design with the exception of using convolution instead of matrix multiplication, as seen in the equations below
- This architecture uses three Convolutional LSTM layers stacked together, each followed by a 3D batch normalization layer, with a final 2D convolutional layer to convert the last hidden state at the last time step to a frame prediction
- Loss function: BCE Loss, optimizer: Adam optimizer, learning rate: 1e-6

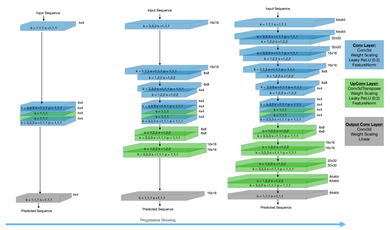


Convolutional LSTM unit:

$$i_t = \sigma(W_{xi} * \mathcal{X}_t + W_{hi} * \mathcal{H}_{t-1} + W_{ci} \circ \mathcal{C}_{t-1} + b_i)$$
$$f_t = \sigma(W_{xf} * \mathcal{X}_t + W_{hf} * \mathcal{H}_{t-1} + W_{cf} \circ \mathcal{C}_{t-1} + b_f)$$
$$\mathcal{C}_t = f_t \circ \mathcal{C}_{t-1} + i_t \circ \tanh(W_{xc} * \mathcal{X}_t + W_{hc} * \mathcal{H}_{t-1} + b_c)$$
$$o_t = \sigma(W_{xo} * \mathcal{X}_t + W_{ho} * \mathcal{H}_{t-1} + W_{co} \circ \mathcal{C}_t + b_o)$$
$$\mathcal{H}_t = o_t \circ \tanh(\mathcal{C}_t)$$

FutureGAN

- GANs reduce blurriness caused by equally-likely future sequences in frame prediction via a Generator *G* that learns to generate frames that Discriminator *D* cannot differentiate from truth. *D* learns to differentiate *G*'s fake frames from truth, and thus learns unlikely sequences
- *G* uses encoder to learn latent representation of past frames, decoder uses this to generate predicted frames. 3D convolutional layers with asymmetric filter size and stride are used to encode and decode spatial and temporal parts of input and up and downsample for progressive growth. Leaky ReLU and pixel-wise normalization are used after convolution in hidden layers
- *D* uses 3D convolutional layers with asymmetric filter size and stride to downsample, and minibatch standard deviation to increase variation in *G*'s output. The fully connected output layer and linear activation function produce the score determining whether an image is fake
- Progressive growth from 4x4 to 128x128 px to stabilize
- Loss function: BCE Loss and Wasserstein GAN with gradient penalty (WGAN-GP) loss with epsilon-penalty (penalty coefficients $\lambda = 10$, $\epsilon = 0.001$), optimizer: Adam optimizer with $\beta_1 = 0$, $\beta_2 = 0.99$, learning rate: 0.001



$$L_D(\boldsymbol{x}, \tilde{\boldsymbol{x}}, \hat{\boldsymbol{x}}) = \underbrace{\mathop{\mathbb{E}}_{\tilde{\boldsymbol{x}} \sim \mathbb{P}_g}[D(\tilde{\boldsymbol{x}})] - \mathop{\mathbb{E}}_{\boldsymbol{x} \sim \mathbb{P}_r}[D(\boldsymbol{x})]}_{\text{WGAN loss}} + \underbrace{\lambda \mathop{\mathbb{E}}_{\hat{\boldsymbol{x}} \sim \mathbb{P}_{\hat{g}}}[(\|\nabla_{\hat{\boldsymbol{x}}} D(\hat{\boldsymbol{x}})\|_2 - 1)^2]}_{\text{gradient-penalty}} + \underbrace{\varepsilon \mathop{\mathbb{E}}_{\boldsymbol{x} \sim \mathbb{P}_r} D(\boldsymbol{x})^2}_{\text{epsilon-penalty}},$$

$$L_G(\tilde{\boldsymbol{x}}) = -\mathop{\mathbb{E}}_{\tilde{\boldsymbol{x}} \sim \mathbb{P}_g}[D(\tilde{\boldsymbol{x}})].$$

$$\tilde{\boldsymbol{x}} = G(\boldsymbol{z}),$$

Experiments

Convolutional LSTM

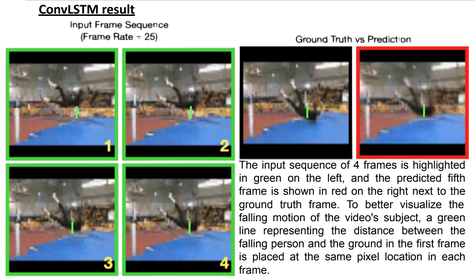
- Trained Convolutional LSTM model on UCF101 subset (365 videos of 4 action classes) for 20 epochs (~60 hrs on NVIDIA T4)
- Using model checkpoint trained on UCF101 subset, trained Convolutional LSTM on full dataset (13.320 videos of 101 action classes) for 7 epochs (~209 hrs on NVIDIA T4)
- Evaluated trained model on UCF101 test set at 25 fps for videos of 64x64 resolution
  - Input sequence of 4 frames, predict 1 frame

GAN:

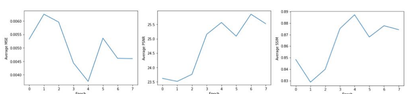
- Trained FutureGAN model on UCF101 subset (365 videos of 4 action classes) for 161 epochs (~66 hrs on NVIDIA T4), using video frames at 128x128 resolution
- Evaluated trained model on UCF101 test subset for videos at 128x128 resolution and 64x64 resolution
- Evaluated trained model on test subset for videos at 25 fps, 12 fps, and 6 fps
  - Input sequence of 6 frames, predict 6 frame

Results & Analysis

ConvLSTM result



The input sequence of 4 frames is highlighted in green on the left, and the predicted fifth frame is shown in red on the right next to the ground truth frame. To better visualize the falling motion of the video's subject, a green line representing the distance between the falling person and the ground in the first frame is placed at the same pixel location in each frame.



Average MSE, PSNR, and SSIM values evaluated at each training iteration over the full test set. At epoch 0, weights trained for 20 epochs over the data subset are used.

FutureGAN result

A sample test video prediction from the FutureGAN model with (a) frame rate of 25 fps, 128x128 px resolution (b) frame rate of 6 fps, 128x128 px resolution (c) frame rate of 25 fps, 64x64 resolution. The input sequence of 6 frames is highlighted in green on the right, and the predicted 6 frames is shown in red on the left under the corresponding ground truth frame in black.

MSE		
Frame	ConvLSTM 64x64	FutureGAN 64x64
1	0.0046	0.0706
2	N/A	0.0795
3	N/A	0.0900
4	N/A	0.0989
5	N/A	0.1062
6	N/A	0.1140
Avg.	0.0046	0.0932

PSNR		
Frame	ConvLSTM 64x64	FutureGAN 64x64
1	25.5266	18.0436
2	N/A	17.5727
3	N/A	17.0052
4	N/A	16.6343
5	N/A	16.3350
6	N/A	16.0261
Avg.	25.8583	16.9361

SSIM		
Frame	ConvLSTM 64x64s	FutureGAN 64x64
1	0.8743	0.3348
2	N/A	0.3086
3	N/A	0.2810
4	N/A	0.2599
5	N/A	0.2424
6	N/A	0.2226
Avg.	0.8777	0.2749

SSIM for FutureGAN 128x128			
Frame	25 fps	12 fps	6 fps
1	0.3011	0.2846	0.2609
2	0.2866	0.2674	0.2431
3	0.2688	0.2507	0.2231
4	0.2553	0.2387	0.2038
5	0.2402	0.2258	0.1915
6	0.2284	0.2138	0.1832
Avg.	0.2634	0.2468	0.2176

MSE for 128x128 FutureGAN			
Frame	25 fps	12 fps	6 fps
1	0.0841	0.0918	0.1038
2	0.0930	0.1047	0.1192
3	0.1031	0.1177	0.1378
4	0.1119	0.1259	0.1586
5	0.1214	0.1352	0.1710
6	0.1289	0.1435	0.1806
Avg.	0.1071	0.1198	0.1452

PSNR for FutureGAN 128x128			
Frame	25 fps	12 fps	6 fps
1	17.4465	17.0446	16.3874
2	17.0350	16.4775	15.8366
3	16.5750	15.9938	15.2262
4	16.2431	15.6777	14.6320
5	15.8703	15.3676	14.3152
6	15.5948	15.0721	14.1123
Avg.	16.4608	15.9389	15.0849

$$MSE = \frac{1}{3mn} \sum_{i=0}^{m-1} \sum_{j=0}^{n-1} \sum_{c=0}^{2} [I(i,j,c) - K(i,j,c)]^2$$

$$PSNR = 10 \log_{10}(\frac{MAX_I^2}{MSE})$$

$$SSIM(x,y) = \frac{(2\mu_x\mu_y + c_1)(2\sigma_{xy} + c_2)}{(\mu_x^2 + \mu_y^2 + c_1)(\sigma_x^2 + \sigma_y^2 + c_2}$$

Quantitative results for ConvLSTM and FutureGAN's 64x64 model evaluated on 25fps, 64x64 px videos, as well as FutureGAN's final 128x128 model evaluated on 25, 12 and 6 fps, 128x128 px videos.

Conclusions & Future Work

Both models achieved reasonable predictions considering their respective training sets, but because ConvLSTM was trained on the full UCF101 dataset, it outperformed FutureGAN quantitatively and qualitatively on the first future frame.

If given unlimited time and compute resources, we would train FutureGAN on the full UCF101 dataset. We would also explore architectural changes, such as the number and design of the intermediate layers in the progressive growth steps. For ConvLSTM, we would modify the number of stacked convolutional LSTM cells and train for more epochs. For both models, we would investigate their long-term predictive abilities by varying the number of input frames and increasing the number of predicted frames. We would also experiment with slower frame rates to evaluate performance on predicting increasingly disjoint motion.

References

[1] X. Shi, Z. Chen, H. Wang, D.-Y. Yeung, W.-k. Wong, and W.-c. Woo, "Convolutional lstm network: A machine learning approach for precipitation nowcast-ing," 2015
[2] S. Aigner and M. Korner, "Futuregan: Anticipating the future frames of video sequences using spatio-temporal 3d convolutions in progressively growing gans," 2018