Self-supervised Video Prediction

Lab CudaVision – Learning Vision Systems on Graphics Cards

Rohil Rao, Ruben Vasquez

Universität Bonn
s6roraoo@uni-bonn.de,s6ruvasq@uni-bonn.de
Matrikelnummers: 3299480, 3340562

Abstract. In this work, we present a deep learning architecture that enables us to predict future frames given past sequences of frames, in a self-supervised and auto-regressive manner. The model is inspired by two encoder-decoder models: Video Ladder Networks (VLN) and RoboCup 2019 AdultSize Winner NimbRo. In this architecture, the pre-trained ResNet-18 model is used as the encoder. The decoder has a series of layers (consisting of batch normalization, convolution, and pixel shuffle layers). The model also has lateral connections from the encoder to the decoder consisting of recurrent connections (ConvGRU) and location-dependent convolutions (LDC). The combination of these elements makes the whole network robust in space and time. As a result, the network performs very well on video prediction tasks. The proposed model is trained and tested on two datasets (Moving MNIST and robot images) and provides satisfactory results.

1 Introduction

In the last few years, we have seen a lot of success in the field of computer vision as a result of advancements in deep learning research particularly in the field of visual recognition. Recently there has been increased interest in numerous research groups related to the difficult task of video prediction. The general objective of video prediction is to build a model that can generate future frames given some sequence of past frames. This task turns out to be quite challenging because the prediction model must be able to learn the internal representations and rules of the world while also understanding the spatial and temporal dependency between frames of the video to make accurate and meaningful predictions.

The significance of the task of video prediction must be noted because it is the motivation of this work. Firstly, for it to be successful, the architecture of a video prediction model should be designed in such a way that it can extract semantic features in both space and time dimensions [Cricri et al., 2016]. Secondly, predicting future frames based on some sequence of past frames can help anticipate the consequences of actions in the real world which could further help autonomous systems to make accurate decisions. Lastly, video prediction has a

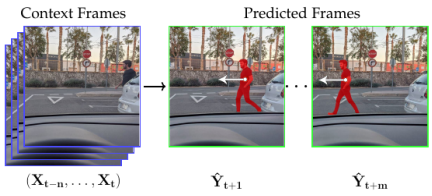

Fig. 1: This picture shows an application of the video prediction model (autonomous driving) [Oprea et al., 2020]. It can be observed that the system takes a set of past frames about the position of a pedestrian as inputs $(X_{t-n}, \ldots, X_t)$ and gives as output the future frames which show predicted movements of the pedestrian (Y_{t+1}, Y_{t+m}). It is noteworthy that this application can be very useful to make decisions. For instance when the car is facing obstacles on the road: either the driver would be alerted by the prediction system or the system could preemptively stop the car to avoid collisions.

broad range of applications such as autonomous driving, robot navigation, and human-machine interaction[Oprea et al., 2020].

The structure of this work will be as follows: The first part is about an introduction along with a formal definition of the task. In the second section, we dive into related work where we explain the related methods and techniques used for video prediction. In the third part, we show the architecture of the model, explaining its components in detail. In the fourth and fifth parts, we explain the training methods we have used as well as the results that we have achieved on the Moving MNIST and Robot dataset respectively. Finally, in the last section, we conclude the work by highlighting certain important observations.

1.1 Problem definition

Let $X_t \in R^{w \times h \times c}$ be the t-th frame in the video sequence $X = (X_{t-n}, \ldots, X_t)$ with n frames, where w, h and c denote width, height, and number of channels, respectively. The target is to predict the next frames $Y = (Y_{t+1}, Y_{t+2}, ..., Y_{t+m})$ from the input X [Oprea et al., 2020].

2 Related Work

This work has been inspired by two models: VLN [Cricri et al., 2016] and NimbRo [Rodriguez et al., 2019].

The first model (VLN) is a neural fully convolutional encoder-decoder network. The network has recurrent and forward lateral connections from the encoder to the decoder at all layers. The model also uses residual blocks inspired

by the ResNet architecture. While the encoder uses dilated convolutional layers and Leaky-Relu activation functions, the decoder uses normal convolutional layers and sigmoid activation functions. Additionally, both encoder and decoder make use of batch normalization. Another interesting thing that has been considered in this architecture is the introduction of a recurrent connection called Convolutional Long Short Memory (ConvLSTM) which is an extension of fully connected LSTM. The advantage of ConvLSTM in the model is to preserve the spatial information.

The second model (NimbRo) was proposed for the RoboCup 2019 competition. This architecture has a visual perception system that is robust to brightness, view angles, and lens distortion. The model uses the pre-trained ResNet-18 as an encoder, due to its strength in visual recognition tasks. The model also makes use of a location-dependent convolutional layer as the final layer to overcome the limitation of spatial invariance of convolutional layers. [Rodriguez et al., 2019].

3 Architecture

The complete architecture as proposed by Hafez Farazi has been represented in 2. The important sections of the architecture are further explained below.

3.1 Location Dependent Convolutions

For video prediction, it is important that the deep learning architecture should be able to analyze the video both spatially and temporally. Due to the location-invariant nature of convolutions, they cannot capture location-dependent features. This architecture makes use of the method of convolutional layers with learnable location-dependent biases and predefined location encodings as proposed by Niloofar Azizi, Hafez Farazi and Behnke, 2018. The proposed method adds learnable location-dependent weights to the convolution layer. Moreover, location-dependent gradients (both the x and y directions of the image) are also added.

3.2 Convolutional Gated Recurrent Unit

The ConvGRU (Convolution Gated Recurrent units) as proposed by Ballas et al. [Learning Video Representations, 2016] are advantageous for learning spatio-temportal features in video frames. The ConvGRU is implemented by replacing the fully connected units in the GRU with convolution operations. This provides the advantage of sparse connections and parameter sharing.

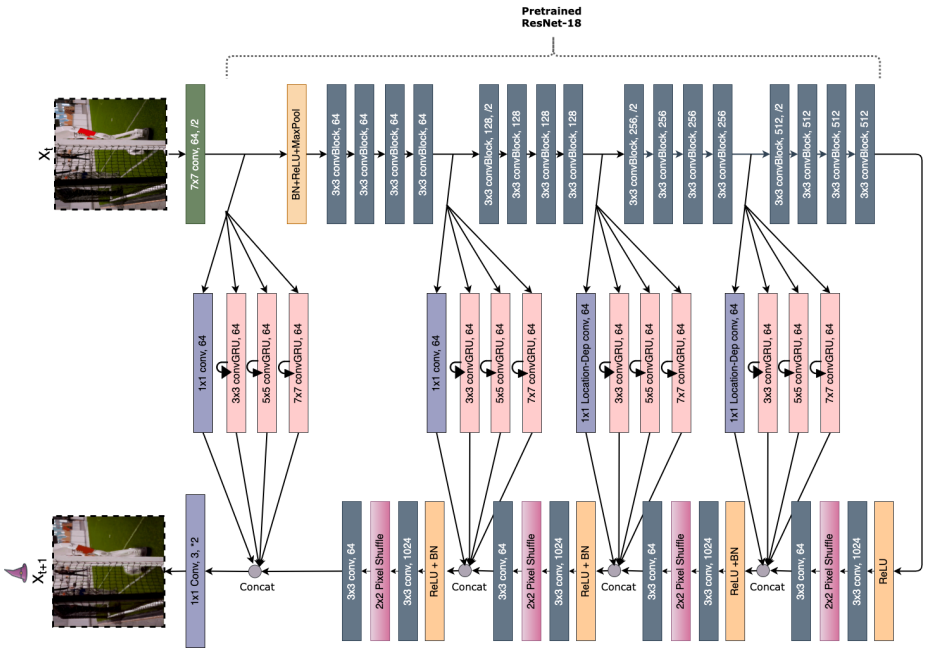

Fig. 2: Self-supervised Video Prediction architecture. This architecture is inspired by the Video Ladder Network and the RoboCup 2019 Winner Architecture NimbRo. Note the architecture uses recurrent ConvGRU cells and location-dependent convolutions.

4 Data preparation

4.1 Moving MNIST

The model was initially tested on the synthetic Moving MNIST dataset for obtaining a baseline performance measure of the proposed architecture on video prediction. The original dataset contains 10,000 sequences each of the length 20 with moving digits. For the purpose of our project we have reduced the sequence length to 6. For checking the correctness of the model, we have only experimented with a few hyper-parameters heuristically, until we obtained convergence on the training and validation dataset.

4.2 Robot data

The performance of the model has also been evaluated on videos of different robots playing soccer at the RoboCup over different years. The dataset has been prepared from videos that are available on YouTube. Some manual preprocessing steps were required for the preparation of the dataset. The frames to be included in this dataset had to be selected manually due to the huge amount

of frames that could be extracted from each video. We found that some chunks of frames were not related and hence removed. We prioritized the collection of frames with robots. Considering memory and resource limitations, the dataset used for training purposes had around 3400 frames obtained from 11 videos. The frames were also resized to have a resolution of 320x224 pixels. Further, the dataset was split into two parts. We used 80% of the data for training and 20% for validation. The dataset was stored as a Numpy file after converting it to the unsigned integer data format to further improve the efficiency of storing and loading the data. The testing dataset was obtained from a separate video similarly. The data used finally for training, validation and testing were randomly sampled from the prepared dataset.

5 Training

5.1 Loss function: DSSIM + L2 Loss

For training we have used a combination of two loss functions: DSSIM Loss and L2 Loss (MSE).

The Structural Dissimilarity Index Measure (DSSIM) is derived from the Structural Similarity Index Measure (SSIM) [Wang et al., 2003] as below:

5.1.1 Mathematical formulation of DSSIM: The DSSIM for two images x and y of the same size can be calculated using SSIM as below:

$$DSSIM(x, y) = \frac{(1 - SSIM(x, y))}{2}$$

5.1.2 Mathematical formulation of SSIM: The SSIM(x,y) for two images x and y of the same size is represented as:

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

In the formula above, for both x and y images, μ and σ represent the mean and the variance of the images and c (c_1 and c_2) is used to denote constants used for mathematical stability.

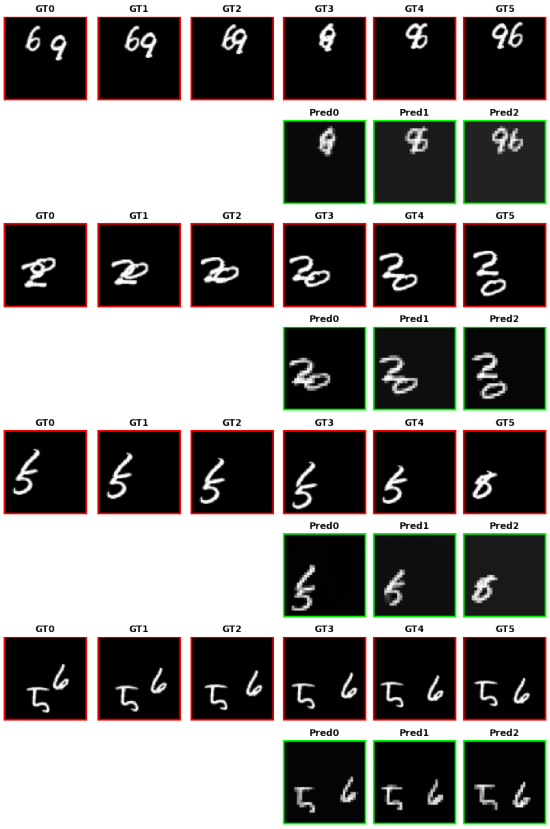

Fig. 3: Sample result on the validation instances for the Self-supervised Video Prediction architecture on the Moving MNIST dataset. The frames bordered in red colour represent the Ground Truth (GT) images and the frames bordered in green color represent the respective Predicted (Pred) images.

5.2 Training on Moving MNIST:

The proposed architecture in Fig.2 is deep and has around 38 million trainable parameters and it is not suitable for the Moving MNIST dataset which is simpler (only black and white images) and much smaller in dimension (64 pixels in width and height). The original architecture in Fig.2 has 8 layers (excluding the output layer) with cross-connections from the encoder to the decoder. We modified the architecture by splitting it into half and reducing the number of layers to 4 (2 each for the encoder and decoder respectively). We preserved the original structure of having cross-connections (consisting of ConvGRU layers and Location Dependent Convolution layers) between all the encoder and decoder layers. The reduced architecture has around 9 million parameters.

The model was trained without the pre-trained weights for the ResNet layers. We used the Adam optimizer and trained for 150 epochs with different learning rates for different layers to ensure good gradient flow through all layers of the network. We used a learning rate of 0.0001 for all layers except the decoder for which we had a learning rate of 0.001. Also, we found that a higher weight (0.8) for the DSSIM loss (with a window size of 3) and a smaller weight (0.2) for the L2 loss produced good-looking output images.

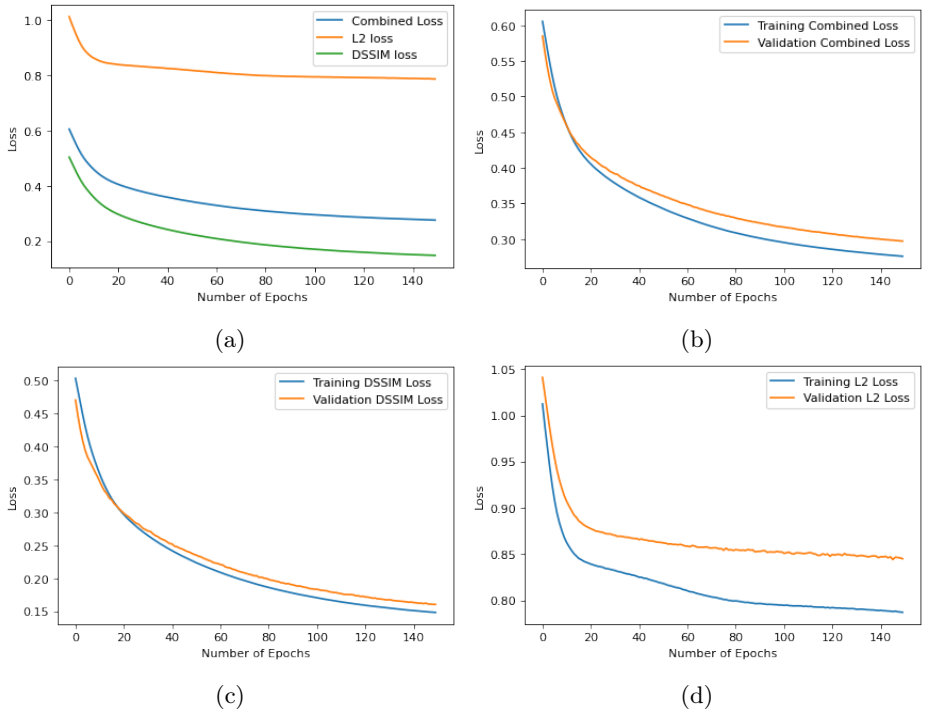

Fig. 4: The figure 4a shows the L2, DSSIM and Combined(L2 and DSSIM) loss for the Moving MNIST training dataset plotted for each epoch. Also figures 4b, 4c ,4d show the Combined, DSSIM and L2 loss for the training and validation dataset respectively plotted for each epoch.

5.2.1 Training on Robot Dataset: The model was trained on around 550 instances which were split into 80% training and 20% validation datasets respectively for around 75 epochs. We also tried to include data augmentation methods like random flipping and changing the hue, saturation, value, and brightness for a small proportion of the image to improve the generalization of the model however this was not included in the final training due to memory issues. In contrast to the Moving MNIST training, we have made use of the pre-trained ResNet-18 weights. We have also experimented heuristically with different learning rates

for different layers of the architecture. We kept a smaller learning rate for the encoder (0.00001) as compared to the decoder(0.0001). As in the case for Moving MNIST, we observed that a higher weight (0.9) for the DSSIM loss (with a window size of 5) and a smaller weight (0.1) for the L2 loss produced good-looking output images.

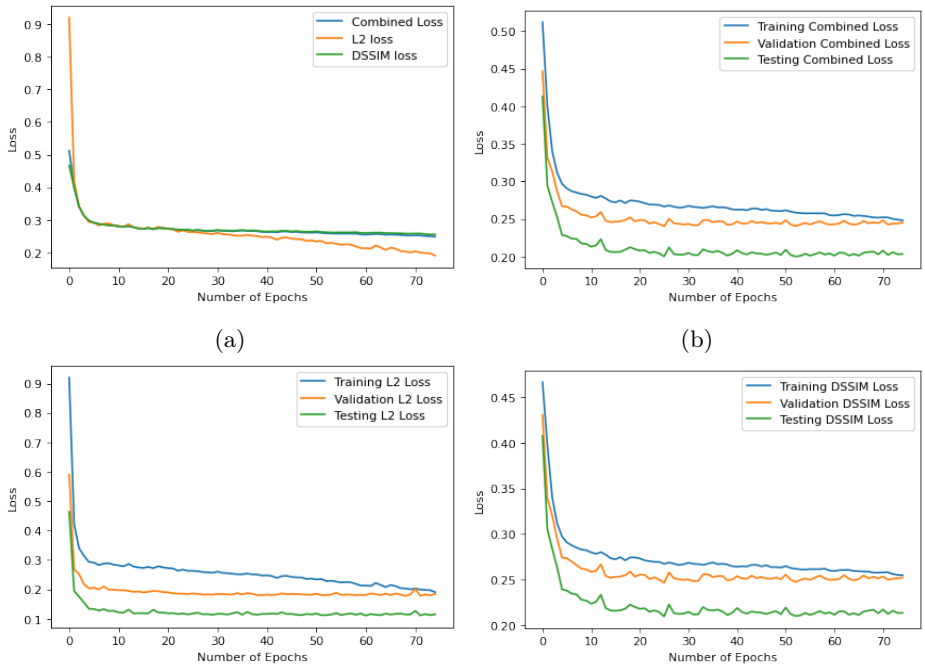

Fig. 5: The figure 5a shows the L2, DSSIM and Combined(L2 and DSSIM) loss for the Robots training dataset plotted for each epoch. Also figures 5b, 5c ,5d show the Combined, L2 and DSSIM loss for the training, validation and testing dataset respectively plotted for each epoch.

6 Results

6.1 Results on Moving MNIST

We obtained good results on the Moving MNIST dataset. We observed convergence on the training and validation datasets after around 150 epochs as seen in Fig.4. The loss metrics for the parameters mentioned above can be seen in table 1. The output images of the predictions of the validation dataset can be seen in Fig.3

losses / Data sets	DSSIM	L2	Combined
Train	0.15	0.8	0.28
Validation	0.16	0.85	0.31

Table 1: Loss metrics on Moving MNIST. Combined loss is calculated with 0.8 weightage to DSSIM and 0.2 weightage to to L2 Loss.

6.2 Results on the Robot Dataset

On the Robot dataset we obtained some satisfactory results. We observed convergence on the training and validation datasets after around 75 epochs as seen in Fig.5. The loss metrics for the parameters mentioned above can be seen in table 2. The output images of the predictions on the testing dataset can be seen in Fig.6

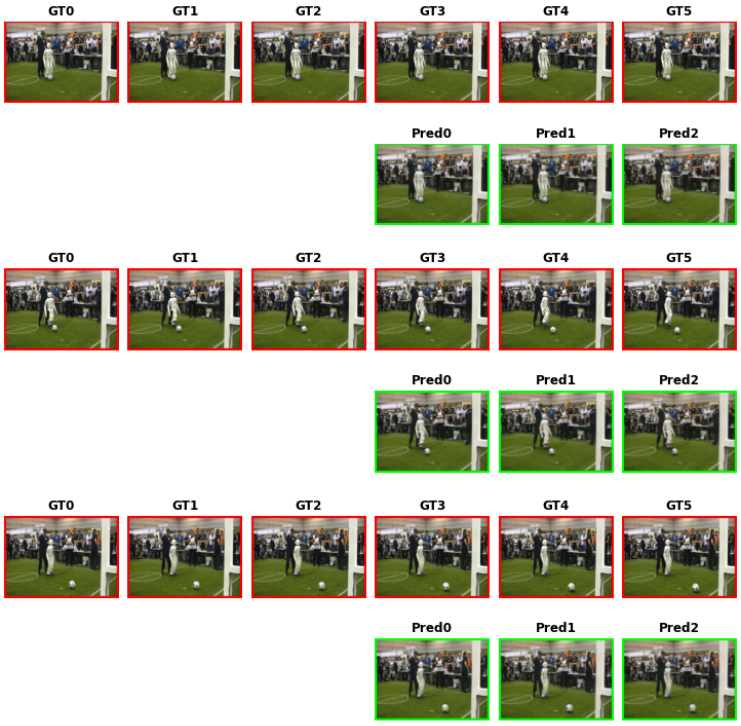

Fig. 6: Sample result on the Testing dataset for the Self-supervised Video Prediction architecture for the Robot dataset. The frames bordered in red colour represent the Ground Truth (GT) images and the frames bordered in green color represent the respective Predicted (Pred) images.

losses / Data sets	DSSIM	L2	Combined
Train	0.27	0.19	0.27
Validation	0.26	0.2	0.26
Test	0.22	0.13	0.21

Table 2: Loss metrics on Robot dataset. Combined loss is calculated with 0.9 weightage to DSSIM and 0.1 weightage to to L2 Loss.

7 Future work

7.1 Inception module

Due to the structure of the network, one possible modification would be to change the lateral connections, from the encoder to the decoder, to resemble the Inception module. Instead of using 3x3, 5x5, and 7x7 convGRU layers directly, we can use a 1x1 convolution before them as done in the Inception module. This can help reduce the number of feature maps.

7.2 Adversarial Training

We can also train the network in an adversarial manner. The basic structure of the original architecture will be preserved. The decoder of the network can be modified to play the role of the Generator. We still use the ResNet-18 encoder, however, it is only connected to the decoder via lateral connections and not directly. Instead of taking an input from the encoder, the decoder (or Generator) will receive a random noise vector as input. We only use the encoder architecture to obtain augmented inputs (that capture spatio-temporal features) at each layer of the decoder (or Generator). The Generator can consist of Transposed Convolutional layers. The Discriminator network can be added at the end of the network.

8 Conclusion

In this work, we have shown that the proposed model is very effective for the task of self-supervised prediction of frames in an auto-regressive manner. We have observed that the various components like the ConvGRU layers and the Location Dependent Convolution layers implemented as lateral connections from the encoder to the decoder make the network robust for spatio-temporal prediction. However, with more time and computing power, there is still a lot of potential to tune the hyperparameters of the model to obtain better results.

9 Acknowledgement

This paper and the associated work are courtesy of Hafez Farazi. We thank him for his support and guidance in the implementation of this work.

References

Cricri, Francesco et al. (2016). "Video ladder networks". In: *arXiv preprint arXiv:1612.01756*.

Learning Video Representations, Delving Deeper into Convolutional Networks for (2016). "Location Dependency in Video Prediction". In: *International Conference on Learning Representations (ICLR)*. Puerto Rico, United States.

Niloofar Azizi, Hafez Farazi and Sven Behnke (2018). "Location Dependency in Video Prediction". In: *International Conference on Artificial Neural Networks (ICANN)*. Rhodes, Greece.

Oprea, Sergiu et al. (2020). "A review on deep learning techniques for video prediction". In: *IEEE Transactions on Pattern Analysis and Machine Intelligence*.

Rodriguez, Diego et al. (2019). "RoboCup 2019 AdultSize winner NimbRo: Deep learning perception, in-walk kick, push recovery, and team play capabilities". In: *Robot World Cup*. Springer, pp. 631–645.

Wang, Zhou et al. (2003). "Multiscale structural similarity for image quality assessment". In: *The Thrity-Seventh Asilomar Conference on Signals, Systems & Computers, 2003*. Vol. 2. Ieee, pp. 1398–1402.