Lab CudaVision: Video Prediction

Amit Kumar Rana[1], Dhagash Desai[2], and Lina Hashem[2]

RWTH Aachen University, University of Bonn

Abstract. Video frames generation is a challenging task due to the wide uncertainty in the nature of the problem. In this lab project, we approach the task of the video prediction using the model discussed in the lab, based on the Video Ladder Network. In our work, the moving **Moving MNIST (MMNIST)** and the **KTH Action** dataset are being used to perform the experiments. We present the effects of various design choices in the model architectures and the training settings. The final results achieved on both datasets are realistic and coherent with the given context frames, indicating the strong learning capability of the network[1].

1 Introduction

The video prediction task refers to generating future frames given some context frames as input. Learning the spatial and temporal dynamics in the video sequences can lead to predicting future frames, which can be used for many downstream tasks such as robot control, reinforcement learning [6], autonomous driving [12], and pedestrian prediction [9] etc. Video prediction is a very challenging task due to the inherent uncertainty in the model dynamics. For example, imagine a ball bouncing off the surface of the floor, after hitting the floor the ball can go in any direction with any velocity based on its hitting direction, the friction of the floor etc.

Encoder-decoder based architectures with Recurrent Neural Networks (RNNs) or Long Short Term Memory Networks (LSTMs) [7] for recurrent connections in the models have been used profusely in a lot of previous works for video prediction [4,9,2,1]. Insipred by Variational Autoencoders (VAEs) [10], some other works have successfully used KL-divergence loss to learn the stochastic nature of temporal dynamics [4,1].

In this project, we are implementing the architecture as discussed in the lab, inspired by the paper Video Ladder Networks [3]. We provide the details of the baseline model and its components in Sec. 2. In Sec. 3, the details of the datasets used, model architectures and training are discussed. We also present ablation analysis in Sec. 4. Qualitative and quantitative results show that even with this kind of simple architecture we are able to generate coherent and sharp future frames given our context frames. Finally, we sum up our report with an outlook for future work in Sec. 6 and a conclusion in Sec. 7.

[1] Project page with all supplementry material and code: https://github.com/Dhagash4/video-prediction

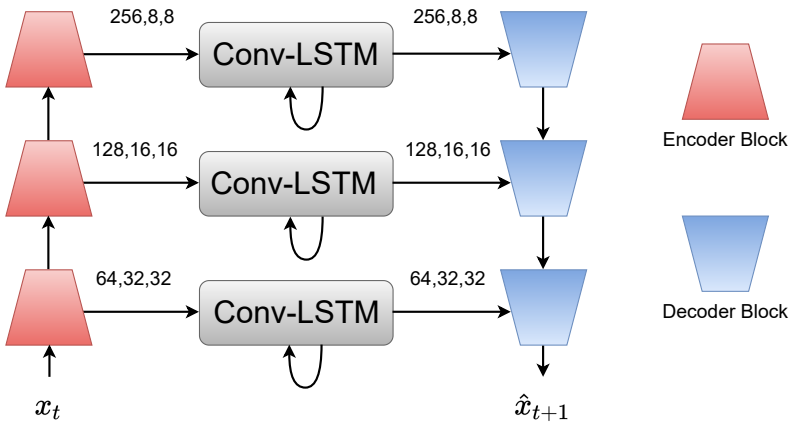

Fig. 1. Baseline Model: Hierarchical baseline model as proposed in the lab, output and input dimensions for corresponding encoder and decoder blocks shown in figure.

2 Method

2.1 Baseline Model

We are using a hierarchical encoder-decoder architecture with recurrent neural networks to learn temporal dynamics, as shown in Fig. 1. As mentioned in [3], the hierarchical approach helps to relieve the burden from higher layers to model lower layers representations.

Encoder and Decoder: The model encoder and decoder both consist of fully connected convolutional blocks in each layer. The encoder blocks each downsample by a factor of two. On the other hand, the decoder is an inverse mirror of the encoder's structure; each block upsamples by a factor of two, and upsampling is done using convolution transpose blocks. Both encoder and decoder use Batch Normalization [8] after convolution layers followed by Leaky ReLU [20] as activation function. In the last layer of the decoder, Leaky ReLU is replaced by Sigmoid as an activation function.

Recurrent Connections: We have used ConvLSTM [17] to model the recurrent connections between encoder and decoder at each layer. Hierarchical recurrent connections help in passing spatial and temporal information from encoder to decoder block at multiple resolutions [3]. The output of recurrent connections and decoder block from the previous layer is concatenated, which then acts as an input to the decoder layer as illustrated in Fig. 1.

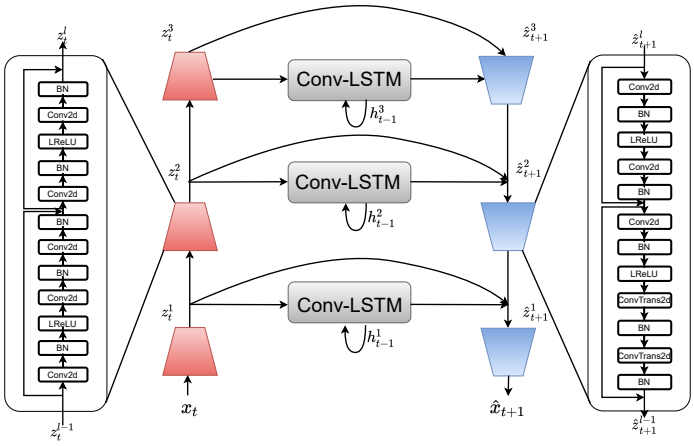

Fig. 2. Extended Model: Extension to baseline model with the addition of skip connections, encoder-decoder blocks are residual blocks.

2.2 Extended Model

Skip Connections: An encoder-decoder architecture consisting of convolutional layers suffers from information loss, especially in upsampling layers in the decoder. Skip connections utilised in [15] became the state of the art method in tackling information loss while dealing with encoder-decoder architectures, helping the decoder in reconstruction with finer details from the bottleneck layer. There are various ways to combine information from the bottleneck layer using skip connections [4,15]. We use a combination of concatenation and convolution for skip connection as used in [3]. Skip connections in video prediction have been proven useful in modelling static information [4].

In our extended model with skip connections, as shown in Fig. 2, the output of the recurrent connection h_t^l, of the feed-forward connection z_t^l and of the upper decoder layer $\tilde{z}_{t+1}^{l+1}$ are merged as follows:

$$\tilde{z}_{t+1}^{l} = LReLU\left(\left(LReLU\left(\left(\tilde{z}_{t+1}^{l+1}, h_t^l\right) * W_h^l\right), z_t^l\right) W_z^l\right) \tag{1}$$

where $LReLU$ is the Leaky ReLU non-linearity, $(.,.)$ denotes channel-wise concatenation, W_h^l and W_z^l are $(1,1)$ convolutional kernel tensors. No batch normalization is applied to the ConvLSTM.

3 Experimental Details

3.1 Datasets

For all our experiments, we are using video sequences of 20 frames where first 10 frames are being used as the context frames and the last 10 frames are predicted future frames. We conducted our experiments on Moving MNIST [18] and KTH Action datasets [16].

Moving MNIST: We are using Moving MNIST, which contains 10,000 sequences each of length 20 showing 2 digits moving in a 64 x 64 frame, as proposed in [18], as our test dataset. The validation dataset has 1000 samples randomly selected from the test dataset. We generated the training dataset on the fly following the method used by [4]. In every epoch, we had 10000 sequences of training samples. The samples were generated by sampling 2 different MNIST [11] digits from the training set (60K total digits). We selected the starting position by uniformly sampling (x,y) as starting locations and the initial velocity vectors were sampled uniformly as $(dx, dy) \in ([-4, 4], [-4, 4])$. After the digits hit a wall the velocity vectors were set to negative of the velocity vectors at the time of impact, as proposed in [4].

KTH: KTH action dataset as proposed in [16], contains 6 types of human actions performed several times by 25 different subjects in four different scenarios. We are using the same configuration as used by [16] to divide our dataset into training, validation and test sets. For generating training sequences, we are using 20 sequences after every 5th frame from the videos. This helps us in increasing the size of the training dataset. The training and test sets have 15010 and 4064 sequences respectively.

3.2 Model Architecture

Our model uses residual blocks for encoder and decoder with the dimensions shown in Fig. 2. We used two conv-LSTM cells at each recurrent connection layer. Detailed analysis for choosing respective model parameters is shown in Sec. 4.

3.3 Training and Criterion

The models were trained using the servers provided by the University of Bonn Informatik Department, where most of the GPU had a capacity of 12GB.

We use mse loss $\mathcal{L}_{rec}$ for reconstruction and LPIPS $\mathcal{L}_{perp}$ for perceptual similarity [21]. The final loss can be modelled as:

$$\mathcal{L} = \mathcal{L}_{rec} + \lambda \mathcal{L}_{perp} \tag{2}$$

where λ is hyperparameter. In our experiments we use $\lambda = 0.4$

We are training in auto-regressive manner and are using teacher forcing, inspired from [4], to train our models, where we feed the ground truth image at each timestamp and generate the image for the next timestamp. Due to the limited availability of computational resources almost all our models and models used for comparison are trained for 100 epochs unless specified otherwise.

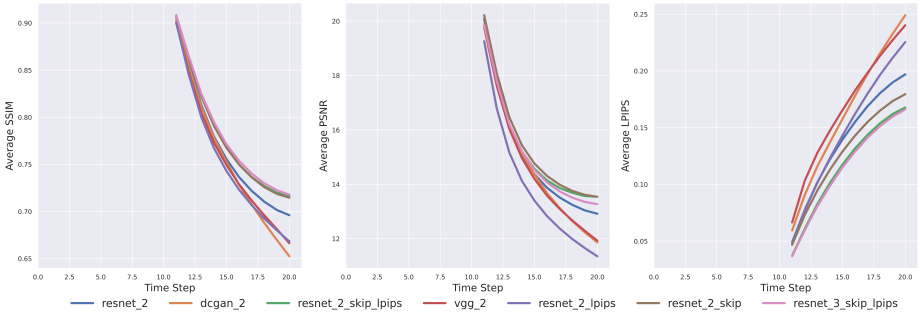

Fig. 3. Ablation Study on MMNIST: Performance metrices shown for different model variants. Model trained with LPIPS loss with residual blocks and skip connections gives best performance.

4 Abalation Studies

In this section, we conduct several experiments by modifying some components of the baseline model, to understand the effectiveness of different components in this model. The metrics used for evaluation are structural similarity (SSIM) [19], Peak Signal-to-Noise Ratio (PSNR) and perceptual metric (LPIPS) [21]. For all these metrics, the average over the total number of samples in the test dataset are displayed in Fig. 3 for the MMNIST dataset. Based on the these metrics the best model is selected, it is worth keeping in mind that these results doesn't fully capture perceptual fidelity thus we also show qualitative results on best performing models in Sec. 5.

Variants: *resnet_2, dcgan_2, vgg_2* in the Fig. 3, refers to the baseline model with encode-decoder blocks based on Resnet [5], DCGAN [14] and VGG-style [13] respectively. The number of conv-LSTM cells in each recurrent connection is 2 and all these models are trained using mse loss criterion. It is evident that the residual blocks based model outperforms all other model variants. It shows the residual blocks are more powerful and have better learning capacity. *dcgan_2* and *vgg_2* have almost similar performances indicating similar learning capacities. *resnet_2_skip* additionally has skip connections from encoder blocks to decoder blocks, compared to *resnet_2*. It can be seen that the model with the skip connections outperforms the one without the skip connection, augmenting the discussion in Sec. 2.2. *resnet_2_lpips* has same architecture as of *resnet_2* but it is trained using the perceptual loss, given in eq. 2. Even though the quantitative metrics show better performance for mse loss, it is shown in Sec. 5, that the qualitative results for perceptual loss are sharper. *resnet_2_skip_lpips* additionally has skip connections from encoder blocks to decoder blocks, compared to *resnet_2_lpips*. *resnet_3_skip_lpips* have 3 conv-LSTM cells in each recurrent connection as compared to *resnet_2_skip_lpips* which has 2. We can see that increasing the number of cells in each recurrent connection to 3 from 2 doesn't

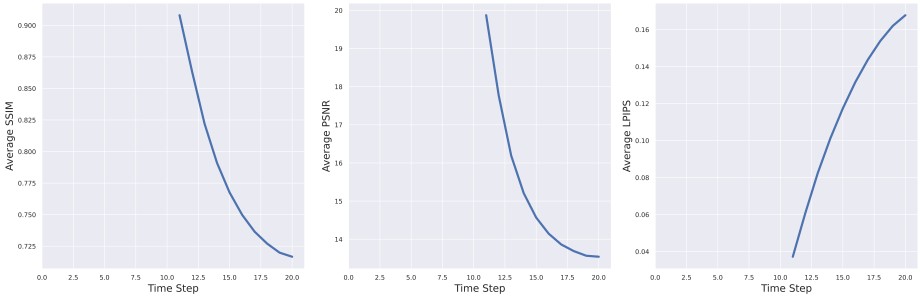

Fig. 4. Quantitative analysis on MMNIST: Performance metrices measured for the best performing model on MMNIST dataset.

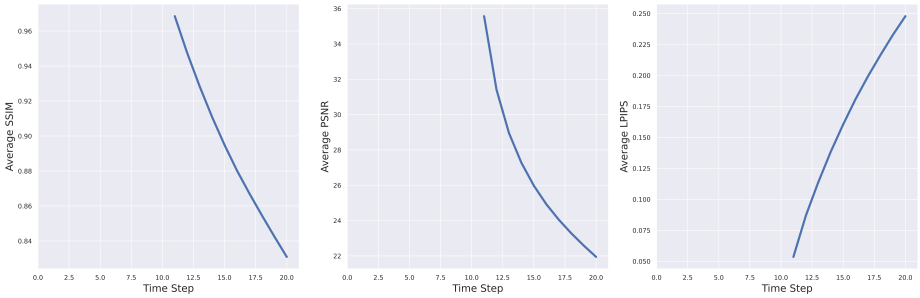

Fig. 5. Quantitative analysis on KTH: Performance metrices measured for the best performing model on KTH dataset.

improve the performance significantly. It increases the number of learnable parameters, hence increasing overall training time. so we take *resnet_2_skip_lpips* as our best model and present the results based on this model in Sec. 5.

5 Results

The following section presents qualitative results for the best models selected after the ablation analysis. We show results corresponding to sequences with the best results on all the metrics discussed in Sec. 4 and also the results on a random batch. Quantitative results for all the model design choices and parameters settings are also presented in this section.

It can be seen in Fig. 6, that for the MMNIST dataset, the images generated are sharp and the model is able to learn the temporal dynamics in the data quite well. For the KTH dataset, as shown in Fig. 7, all the sequences corresponding to best metrics have no motion in the future frames and hence only the static information is reconstructed with very high precision. It is evident from Fig. 7, the model is not able to learn the temporal dynamics very well. It can be because the model is deterministic and it's hard for the model to learn the complex

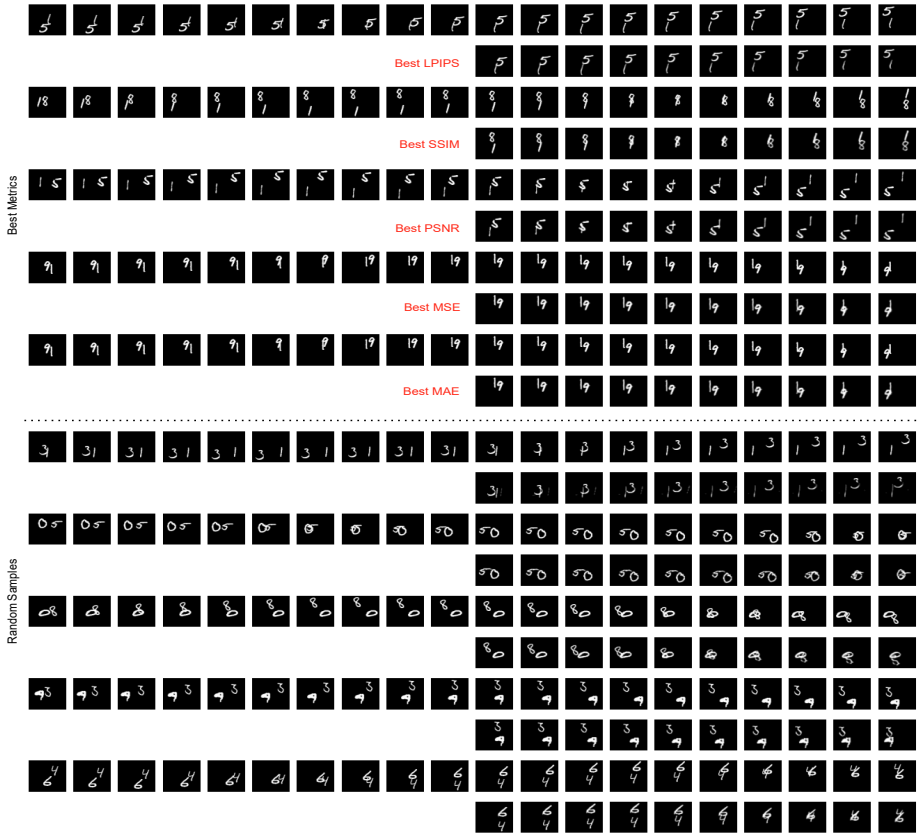

Fig. 6. Qualitative analysis on MMNIST: The line with 20 frames shows ground truth sequences and the following line has the predicted frames. **Best Metrices:** Sequences from test datset with best performances on respective metrices. **Random Samples:** Output for random samples selected from test dataset.

temporal dynamics of KTH dataset. Also. we could only train our models for 200 epochs which is very less for video prediction task.

Fig. 4 and Fig. 5 display the avergage SSIM, PSNR and LPIPS computed on test datasets for the MMNIST and KTH datasets using theirs best performing models respectively.

6 Outlook

As part of our project, we experimented with various encoder-decoder architectures and various design choices for the parameters and training process. Stochastic nature in video-prediction is quite complicated to model. Ideally, the models are trained for a much longer duration approximately 1000 epochs [3].

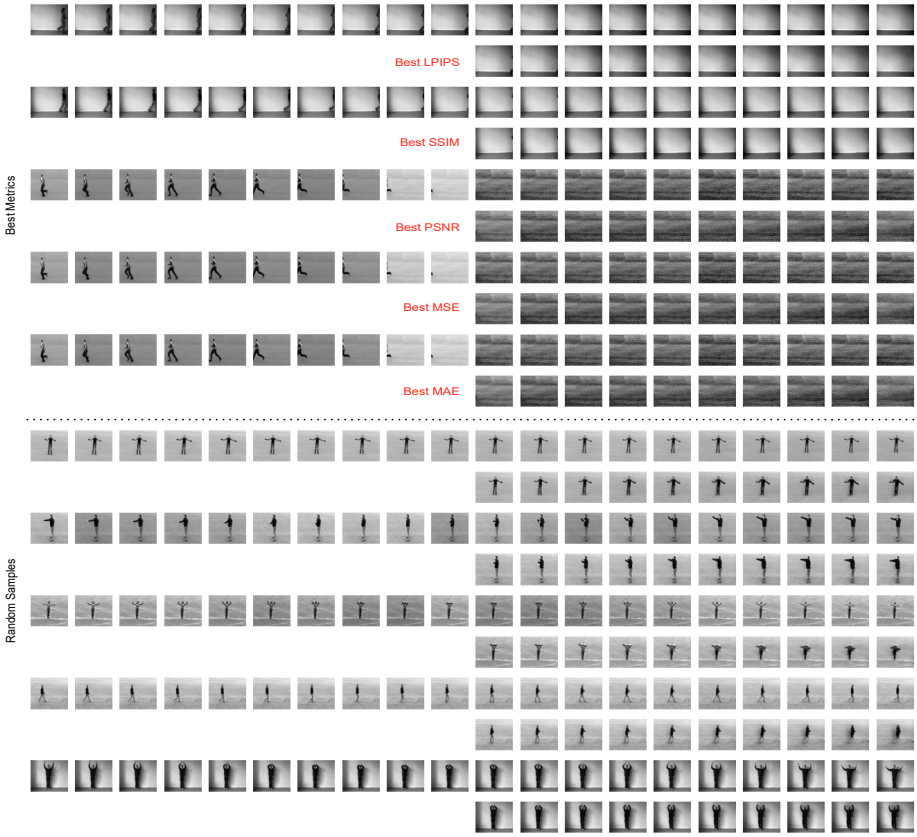

Fig. 7. Qualitative analysis on KTH: The line with 20 frames shows ground truth sequences and the following line has the predicted frames. **Best Metrices:** Sequences from test datset with best performances on respective metrices. **Random Samples:** Output for random samples selected from test dataset.

We believe training it for longer duration will result in better qualitative and quantitative results on both the datasets. As mentioned in Sec. 5, the model fails in learning highly complex temporal dynamics of KTH dataset As shown in [4], we can use the idea of generating stochastic latent variable at all the hierarchies. Also, we have not evaluated different recurrent connections that can be used instead of convolution LSTMs.

7 Conclusion

We have implemented the baseline model introduced in this lab and experimented with it using various design choices. We presented a detailed analysis of various design choices and training settings. Based on that, we came up with an

extension of the baseline model, which is shown to be the best performing model. The simple framework proposed is sufficiently able to learn the temporal dynamics for the synthetic Moving MNIST dataset as well as for real world KTH action dataset, thus generating high-quality predictions on both datasets. We have also mentioned the further improvements to extend the proposed model.

References

1. Mohammad Babaeizadeh, Chelsea Finn, Dumitru Erhan, Roy H. Campbell, and Sergey Levine. Stochastic variational video prediction. In *International Conference on Learning Representations*, 2018.
2. Lluis Castrejon, Nicolas Ballas, and Aaron Courville. Improved conditional vrnns for video prediction. In *The IEEE International Conference on Computer Vision (ICCV)*, October 2019.
3. Francesco Cricri, Xingyang Ni, Mikko Honkala, Emre B. Aksu, and M. Gabbouj. Video ladder networks. *ArXiv*, abs/1612.01756, 2016.
4. Emily Denton and Rob Fergus. Stochastic video generation with a learned prior. *CoRR*, abs/1802.07687, 2018.
5. Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 770–778, 2016.
6. Yung-Han Ho, Chuan-Yuan Cho, and Wen-Hsiao Peng. Deep reinforcement learning for video prediction. In *2019 IEEE International Conference on Image Processing (ICIP)*, pages 604–608, 2019.
7. Sepp Hochreiter and Jürgen Schmidhuber. Long short-term memory. *Neural computation*, 9:1735–80, 12 1997.
8. Sergey Ioffe and Christian Szegedy. Batch normalization: Accelerating deep network training by reducing internal covariate shift, 2015.
9. Christoph G. Keller and Dariu M. Gavrila. Will the pedestrian cross? a study on pedestrian path prediction. *IEEE Transactions on Intelligent Transportation Systems*, 15(2):494–506, 2014.
10. Diederik P Kingma and Max Welling. Auto-encoding variational bayes, 2013.
11. Y. Lecun, L. Bottou, Y. Bengio, and P. Haffner. Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 86(11):2278–2324, 1998.
12. Jesse Levinson, Jake Askeland, Jan Becker, Jennifer Dolson, David Held, Soeren Kammel, J. Zico Kolter, Dirk Langer, Oliver Pink, Vaughan Pratt, Michael Sokolsky, Ganymed Stanek, David Stavens, Alex Teichman, Moritz Werling, and Sebastian Thrun. Towards fully autonomous driving: Systems and algorithms. In *2011 IEEE Intelligent Vehicles Symposium (IV)*, pages 163–168, 2011.
13. Shuying Liu and Weihong Deng. Very deep convolutional neural network based image classification using small training sample size. In *2015 3rd IAPR Asian Conference on Pattern Recognition (ACPR)*, pages 730–734, 2015.
14. Alec Radford, Luke Metz, and Soumith Chintala. Unsupervised representation learning with deep convolutional generative adversarial networks, 2015.
15. Olaf Ronneberger, Philipp Fischer, and Thomas Brox. U-net: Convolutional networks for biomedical image segmentation, 2015.
16. C. Schuldt, I. Laptev, and B. Caputo. Recognizing human actions: a local svm approach. In *Proceedings of the 17th International Conference on Pattern Recognition, 2004. ICPR 2004.*, volume 3, pages 32–36 Vol.3, 2004.

17. Xingjian Shi, Zhourong Chen, Hao Wang, Dit-Yan Yeung, Wai-kin Wong, and Wang-chun Woo. Convolutional lstm network: A machine learning approach for precipitation nowcasting, 2015.
18. Nitish Srivastava, Elman Mansimov, and Ruslan Salakhutdinov. Unsupervised learning of video representations using lstms. *CoRR*, abs/1502.04681, 2015.
19. Zhou Wang, A.C. Bovik, H.R. Sheikh, and E.P. Simoncelli. Image quality assessment: from error visibility to structural similarity. *IEEE Transactions on Image Processing*, 13(4):600–612, 2004.
20. Bing Xu, Naiyan Wang, Tianqi Chen, and Mu Li. Empirical evaluation of rectified activations in convolutional network, 2015.
21. Richard Zhang, Phillip Isola, Alexei A Efros, Eli Shechtman, and Oliver Wang. The unreasonable effectiveness of deep features as a perceptual metric. In *CVPR*, 2018.