iJRASET
International Journal For Research in
Applied Science and Engineering Technology

Volume: 10 Issue: V Month of publication: May 2022

DOI: https://doi.org/10.22214/ijraset.2022.42734

Subsequent Frame Prophecy

Manav Diwan[1], Shyam Taparia[2], Kashish Tayal[3], Sakshi Jha[4]
[1, 2, 3]B.Tech. Scholar, Maharaja Agrasen Institute Of Technology, Rohini, Delhi
[4]Assistant Professor, Department of Computer Science and Engineering, MAIT

Abstract: Future frame prediction project aims at predicting future frames of a video given previous frames. If 'n' frames are given into the model (n being 19 in our case) our model predicts the n+1th frame in the sequence (i.e., 20th frame). The model used for prediction is a deep learning model - GAN (Generative Adversarial Model). The Generative adversarial model has 2 components: the generator which generates the 20th frame and the adversary (Critic) which compares the outputs of the generator with real outputs.

The purpose of this comparison is to train both the generator and the critic in a cyclic fashion to the point where the generator can create almost real-looking outputs. Complex systems like aiming machines, self-driving cars, etc require a good level of correct future prediction to make their outputs correct. We aim at creating a common prediction model which can be generalized for any task and can be used in any such machine.

I. INTRODUCTION

The project involves developing a Neural Network model that can predict the next frame of a video. Intelligent decision- making systems require the capacity to foresee, anticipate, and reason about future events. Auto driving systems, Articulated robots, Aiming Systems, etc. are some of the many intelligent decision-making systems which require a good sense of possible future predictions to output correct and usable predictions in their corresponding fields. The roots of this project are based on one of the coherent features of human intelligence i.e., to predict the future. Future prediction,here, does not mean having the powers of Doctor Strange to see all possible futures at a long period. Humans have a conceptual sense of the future and can predict a possible version of it given present and past information.

The human brain is able to do so by continuously interacting with its environment and remembering how similar objects react in previously understood environments in the presence of a general sense of the laws of nature/physics. For example- In a game of catch, if Person A throws a ball to Person B then without knowing the weight of the ball, air resistance, gravitational pull of the area, etc. but with enough practice, person B can predict an approximate place to be to catch theball. Several computer games like Call of Duty, Dota 2, etc. are entirely built to be won by the person who can predict the next move of their opponent better, under the distinct rules of that game.

Thus, we aim to build a Neural Network modelto be able to artificially replicate this ability of humans to predict future frames of a given video given past frames. The model will focus on predicting the frames with the best quality and attention to detail. Eventually, as the past frames of the video are used to predict future frames, the number of previously predicted frames will start to increase in the framesthat will be used to predict further frames. This is when most previous models start to degrade in the quality of their prediction. Hence, we aim to build a new architecture for predicting the frames better, both using actual frames and predicted frames.

We will be using a deep learning model in our project - GANs (Generative Adversarial Networks) with Wasserstein Loss. There are 2 components in the model - Generator and Critic. The role of the Generator is to predict the next frame in the sequence given previous frames and random noise. The critic also takes in the previous frames but it also takes the outputof the Generator as the input and creates a score in the range of (-Infinity, Infinity) which is used to train both these modelstill the Generator is successfully able to fool the Critic in as much that the Critic cannot differentiate the Generators outputsfrom the real frames.

The Dataset that we are using is the Moving MNIST dataset. This is the moving variant of the MNIST database of handwritten digits. This is the data used by the authors for reporting model performance. The Moving MNIST dataset contains 10,000 video sequences, each consisting of 20 frames. In each video sequence, two digits move independently around the frame, which has a spatial resolution of 64×64 pixels.

The evaluation in Video datasets is both visual and technical. The visual evaluation is the easiest way to see and comparethe actual and predicted next frames in the sequence. Along with this technical approaches used to compare the results of the model are - Mean Squared Error (MSE) and Structural Similarity Index (SSIM).

II. LITERATURE SURVEY

A. Abstract Of Paper Referred

1) The ability to predict, anticipate and reason about future outcomes is a key component of intelligent decision-makingsystems. In light of the success of deep learning in computer vision, deep-learning-based video prediction emerged as a promising research direction. Defined as a self-supervised learning task, video prediction represents a suitable frameworkfor representation learning, as it demonstrates potential capabilities for extracting meaningful representations of the underlying patterns in natural videos. Motivated by the increasing interest in this task, we provide a review of the deep learning methods for prediction in video sequences. They firstly define the video prediction fundamentals, as well as mandatory background concepts and the most used datasets. Next, They carefully analyze existing video prediction models organized according to a proposed taxonomy, highlighting their contributions and their significance in the field. The summary of the datasets and methods is accompanied by experimental results that facilitate the assessment of the state of the art on a quantitative basis. The paper is summarized by drawing some general conclusions, identifying open research challenges, and by pointing out future research directions. They find that exposure to temporal sequences improves the prediction of future events. Learning to predict from temporal sequences generalizes to untrained stimuli. Learning to predict is sensitive to the global structure of the trained sequence. Learning to predict is compromised by increased attention load.

2) Prediction errors (PE) are a central notion in theoretical models of reinforcement learning, perceptual inference, decision-making, and cognition, and prediction error signals have been reported across a wide range of brain regions andexperimental paradigms. They attempted to see the forest for the trees and consider the commonalities and differences of reported PE signals in light of suggestions that the computation of PE forms a fundamental mode of brain function. They suggested that while the encoding of PE is a common computation across brain regions, the content and function ofthese error signals can be very different and are determined by the afferent and efferent connections within the neural circuitry in which they arise.

3) They use long short-term memory (lstm) networks to learn representations of video sequences. Their model uses an encoder lstm to map an input sequence into a fixed-length representation. This representation is decoded using single ormultiple decoder lstms to perform different tasks, such as reconstructing the input sequence or predicting the future sequence. They experiment with two kinds of input sequences - patches of image pixels and high level representations ("percepts") of video frames extracted using a pre-trained convolutional net. They explore different design choices such as whether the decoder lstms should condition on the generated output. They analyse the outputs of the model qualitatively to see how well the model can extrapolate the learned video representation into the future and into the past.They further evaluate the representations by finetuning them for a supervised learning problem - human action recognition on the ucf-101 and hmdb-51 datasets. They show that the representations help improve classification accuracy, especially when there are only a few training examples. Even models pretrained on unrelated datasets (300 hours of YouTube videos) can help action recognition performance. In order to autonomously learn wide repertoires of complex skills, robots must be able to learn from their own autonomously collected data, without human supervision. One learning signal that is always available for autonomously collected data is a prediction: if a robot can learn to predictthe future, it can use this predictive model to take actions to produce desired outcomes, such as moving an object to a particular location. However, in complex open-world scenarios, designing a representation for prediction is difficult. In this work, they instead aim to enable self-supervised robotic learning through direct video prediction: instead of attempting to design a good representation, they directly predict what the robot will see next, and then use this model toachieve desired goals. A key challenge in video prediction for robotic manipulation is handling complex spatial arrangements such as occlusions. To that end, they introduced a video prediction model that can keep track of objects through occlusion by incorporating temporal skip-connections. Together with a novel planning criterion and action space formulation, they demonstrate that this model substantially outperforms prior work on video prediction-based control. Their results show manipulation of objects not seen during training, handling multiple objects, and pushing objects around obstructions. These results represent a significant advance in the range and complexity of skills that can be performed entirely with self-supervised robotic learning.

B. Datasets Explored

1) Action and Human Pose Recognition Datasets

a) KTH: KTH is an action recognition dataset that includes 2391 video sequences of 4 seconds mean duration, each of them containing an actor performing an action taken with a static camera, over homogeneous backgrounds, at 25 frames per second (fps) and with its resolution down sampled to 160 120 pixels. Just 6 different actions were performed, but it was the biggest dataset of this kind at the moment.

b) Hmdb-51: Hmdb-51 is a large-scale database for human motion recognition. It claims to represent the richness of humanmotion taking profit from the huge amount of video available online. It is composed of 6766 normalized videos(with a mean duration of 3.15 seconds) where humans appear performing one of the 51 considered action categories.

c) Ucf101: Ucf101: is an action recognition dataset of realistic action videos, collected from YouTube. It has 101 differentaction categories, and it is an extension of ucf50, which has 50 action categories.

2) Video Prediction

a) Standard Bouncing Balls Dataset: A common test set for models that generate high dimensional sequences. It consists of simulations of three balls bouncing in a box. Its clips can be generated randomly with custom resolution but the common structure is composed of 4000 training videos, 200 testing videos, and 200 more for validation. These kinds of datasets are purely focused on video prediction. Van hateren dataset of natural videos: is a very small dataset of 56 videos, each 64 frames long, that has been widely used in unsupervised learning.

b) Norb Videos: Norb videos dataset is a compilation of static stereo pairs of 50 homogeneously colored objects from various points of view and 6 lightning conditions. Those images were processed to obtain their object masks and even their casted shadows, allowing them to augment the dataset by introducing random backgrounds. Viewpointsare determined by rotating the camera through 9 elevations and 18 azimuths (every 20 degrees) around the object. Norb videos' dataset was built by sequencing all these frames for each object.

c) Moving MNIST: Moving MNIST (m-mnist): is a video prediction dataset built from the composition of 20-frame video sequences where two handwritten digits from the mnist database are combined inside a 64 64 patch, and moved with some velocity and direction along with frames, potentially overlapping between them. This dataset is almost infinite (as new sequences can be generated on the fly), and it also has interesting behaviorsdue to occlusions and the dynamics of digits bouncing off the walls of the patch. For these reasons, this datasetis widely used by many predictive models. A stochastic variant of this dataset is also available. In the original m-mnist, the digits move with constant velocity and bounce off the walls in a deterministic manner. In contrast, in sm-mnist digits move with a constant velocity along a trajectory until they hit a wall at which pointthey bounce off with a random speed and direction. In this way, 8 moments of uncertainty (each time a digithits a wall) are interspersed with deterministic motion.

3) Other Purpose or multi-purpose Datasets

a) Camvid
b) Caltech pedestrian dataset
c) Kitti
d) Cityscapes
e) Ai steering angle
f) Visor
g) Prost
h) Arcade learning environment
i) Inria 3d movie dataset v2
j) Robotrix
k) Uasol
l) Youtube-8m

III. RESEARCH METHODOLOGY

A. Objectives to be Achieved

1) To research the existing techniques and analyze the achievements already made in the field.
2) We will look into the existing techniques used to successfully predict the new frames of a video comparable tothe best results existing at the time.
3) To set a new milestone in the field of video prediction for other fellow researchers to follow.
4) To successfully implement new techniques for the task.
5) We will look into new and improved techniques like GANs, StyleGANs, Variational Encoders, Recurrent NeuralNetworks and try to implement them for the task.

6) To analyze and improve the implementation of the latest techniques.

7) We will try to tune the implementation of the latest techniques to match our requirements and find the bestpossible result from the best possible model given the information from the analysis.

8) To find a good and established result with the techniques used.

9) We will try to find the best and stable results from the techniques comparing the multiple approaches andchoosing the best one we could find.

B. Solution

1) We have used a Deep Learning approach through Generative Adversarial Networks(GANs). We have implemented a basic Generator - Critic architecture using Pytorch and through the use of the Moving MNIST(MMNIST) dataset, trained the model.

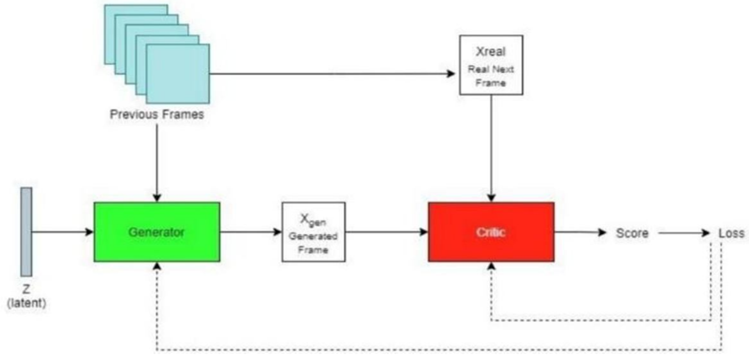
2) The architecture of our model is a typical Generative Adversarial Network(GAN), with a generator-critic system. Videos are passed into through a convolutional 3D.

3) The Generator consists of Convolutional 3D blocks made using batch normalization and activation layers. The Critic is also made using Convolutional 3D blocks but with some fully connected layers at the end.

4) The Generator takes in all of the frames as a 3D image where the channels represent the 19 frames and creates a new frame as the prediction, the next frame.

5) The critic then looks at all the previous frames and the created one and the real one to give out a score for the respective frame's reality with the previous frames. This score is then used to calculate losses and optimize the generator and critic in the respective job using an optimizer, currently Adam.

6) The loss used is called Wasserstein Loss which is just a simplified Earth-Mover's Distance function.



C. Software Specification

1) *Model:* The main core of the project is the Neural Network Model - GAN which is used for the predictions. The model is developed in Python with PyTorch and other supporting libraries. But the crux of the project is entirely built usingPyTorch. Besides that, libraries like NumPy, Pandas, Urllib, Pathlib, etc are used as well.

a) *PyTorch:* PyTorch is an open-source machine learning library based on the Torch library, used for applications such as computer vision and natural language processing, primarily developed by Facebook's AI Research lab. It is free and open-source software released under the Modified BSD license. PyTorch defines a class called Tensor (torch.Tensor) to store and operate on homogeneous multidimensional rectangular arrays of numbers. PyTorch Tensors are similar to NumPy Arrays, but can also be operated on a CUDA-capable Nvidia GPU. PyTorch supportsvarious sub-types of Tensors. PyTorch is built on the Python and torch library which supports computations of tensors on Graphical Processing Units. Tensors are the workhorse of PyTorch. We can think of tensors as multi- dimensional arrays. PyTorch has an extensive library of operations on them provided by the torch module. PyTorch Tensors are very close to the very popular NumPy arrays. In fact, PyTorch features seamless interoperability with NumPy. Compared with NumPy arrays, PyTorch tensors have the added advantage that bothtensors and related operations can run on the CPU or GPU. The second important thing that PyTorch provides allows tensors to keep track of the operations performed on them 35 that helps to compute gradients or derivatives of an output with respect to any of its inputs.

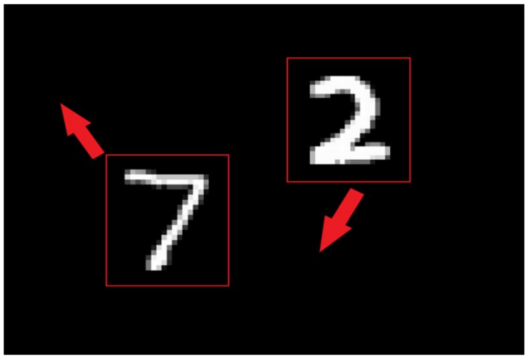
b) Torchvision: The torchvision package consists of popular datasets, model architectures, and common image transformations for computer vision. Torchvision is a library for Computer Vision that goes hand in hand with PyTorch. It has utilities for efficient Image and Video transformations, some commonly used pre-trained models, and some datasets. Torchvision Transforms are common image transformations. They can be chained together using Compose. Most transform classes have a function equivalent: functional transforms give fine-grained control overthe transformations. This is useful if you have to build a more complex transformation pipeline (e.g. in the case of segmentation tasks). Most transformations accept both PIL images and tensor images, although some transformations are PIL-only and some are tensor-only. The Conversion Transforms may be used to convert to and from PIL images. The transformations that accept tensor images also accept batches of tensor images. A Tensor Image is a tensor with (C, H, W) shape, where C is a number of channels, H and W are image height and width. A batch of Tensor Images is a tensor of (B, C, H, W) shape, where B is a number of images in the batch. Torchvision.utils.make_grid() returns a tensor which contains the grid of images. It is a handy function to visualizethe batched outputs of a model.

2) Numpy: NumPy, which stands for Numerical Python, is a library consisting of multidimensional array objects and a collection of routines for processing those arrays. Numpy can be used both for Mathematical computations as well as loading certain types of data files. Pur Moving MNIST dataset has an extension of npy, which is a data format built using Numpy so it can be written and read, only using Numpy. Using NumPy, mathematical and logical operations on arrays can be performed. NumPy is a Python package. It stands for ‘Numerical Python. It isa library consisting of multidimensional array objects and a collection of routines for processing of array. The most important object defined in NumPy is an N-dimensional array type called ndarray. It describes the collectionof items of the same type. Items in the collection can be accessed using a zero-based index. Every item in a ndarray takes the same size as the block in the memory. Each element in ndarray is an object of the data-type object (called dtype). Any item extracted from the ndarray object (by slicing) is represented by a Python object of one of the array scalar types

3) Matplotlib: Matplotlib is a plotting library for the Python programming language and its numerical mathematics extension NumPy. It provides an object-oriented API for embedding plots into applications using general-purpose GUI toolkits like Tkinter, wxPython, Qt, or GTK. There is also a procedural "pylab" interface based on a state machine(like OpenGL), designed to closely resemble that of MATLAB, though its use is discouraged. Matplotlib was originally written by John D. Hunter. Since then it has had an active development community and been distributed under a BSD-style license. Michael Droettboom was nominated as matplotlib's lead developer shortly before John Hunter's death in August 2012 and was further joined by Thomas Caswell.

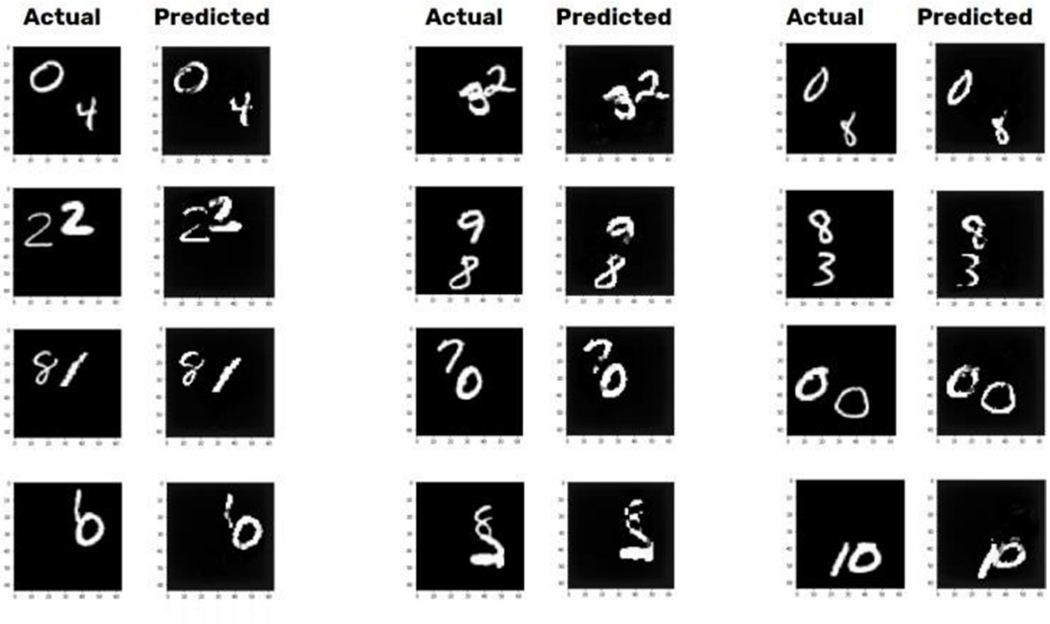
D. Dataset

The only dataset used in our project is the Moving MNIST dataset. Moving MNIST (M-MNIST)[7]: is a video predictiondataset built from the composition of 20-frame video sequences where two handwritten digits from the MNIST database are combined inside a 64 ×64 patch, and moved with some velocity and direction along with frames, potentially overlapping between them. This dataset is almost infinite (as new sequences can be generated on the fly), and it also has interesting behaviors due to occlusions and the dynamics of digits bouncing off the walls of the patch. For these reasons, this dataset is widely used by many predictive models. A stochastic variant of this dataset is also available. In the original M-MNIST, the digits move with constant velocity and bounce off the walls in a deterministic manner. In contrast, in SM-MNIST digits move with a constant velocity along a trajectory until they hit a wall at whichpoint they bounce off with a random speed and direction. In this way, moments of uncertainty (each time a digit hitsa wall) are interspersed with deterministic motion

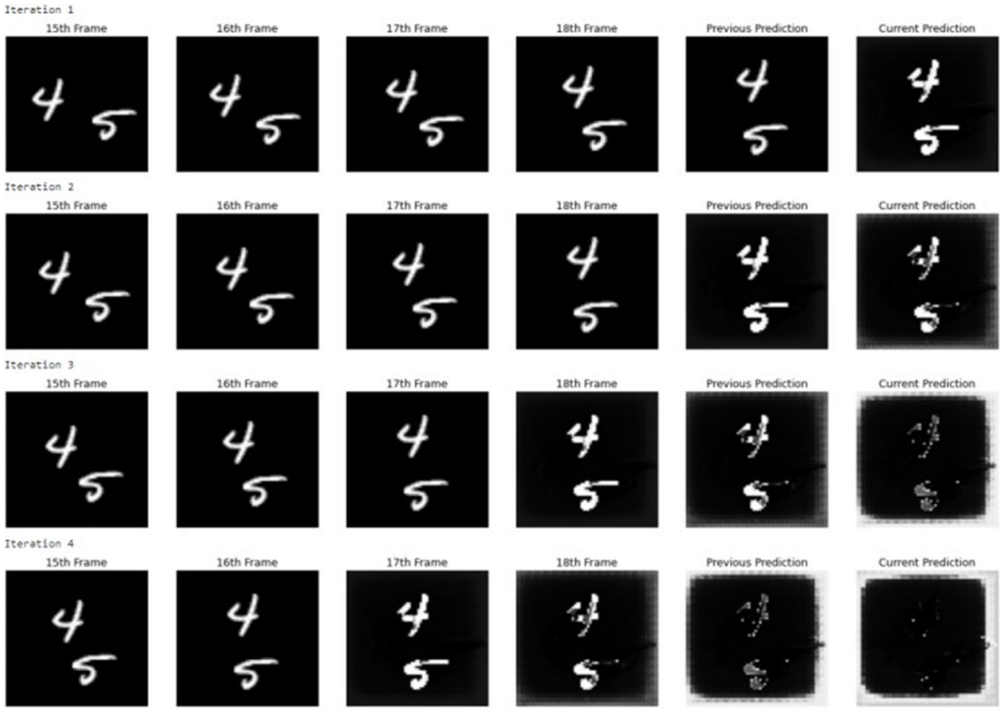


IV. EXPERIMENTAL RESULTS

Since the most widely used evaluation protocols for video prediction rely on image similarity-based metrics, the need for fairer evaluation metrics is imminent. A fair metric should not penalize predictions that deviate from the ground truth at the pixel level, if their content represents a plausible future prediction in a higher level, i.e., the dynamics of the scene correspond to the reality of the labels. In this regard, some methods evaluate the similarity between distributions or at ahigher level. How- ever, there is still room for improvement in the evaluation protocols for video prediction and generation.



Result comparison



Model result over iterations

A. Evaluation Metric Used

The mean squared error (MSE) tells you how close a regression line is to a set of points. It does this by taking the distances from the points to the regression line (these distances are the "errors") and squaring them. The squaring is necessary to remove any negative signs. It also gives more weight to larger differences. It's called the mean squared error as you're finding the average of a set of errors. The lower the MSE, the better the forecast. It's important to note that a value of 0 for MSE indicates perfect similarity. A value greater than one implies less similarity and will continue togrow as the average difference between pixel intensities increases as well.

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

When comparing images, the mean squared error (MSE)–while simple to implement–is not highly indicative of perceivedsimilarity. Structural similarity aims to address this shortcoming by taking texture into account. The results have been calculated after training the Model for 1250 steps with a batch size of 8.

MSE is 0.21064980461567206 mean for a total of 1510 different predictions.

The Structural Similarity Index (SSIM) metric extracts 3 key features from an image:

1) *Luminance:* Luminance is measured by averaging over all the pixel values. Its denoted by µ (Mu) and the formula is given below,

$$\mu_x = \frac{1}{N}\sum_{i=1}^{N} x_i.$$

2) *Contrast:* It is measured by taking the standard deviation (square root of variance) of all the pixel values. It is denoted by σ (sigma) and represented by the formula below

$$\sigma_x = \left(\frac{1}{N-1}\sum_{i=1}^{N}(x_i - \mu_x)^2\right)^{\frac{1}{2}}$$

1) *Structure:* The structural comparison is done by using a consolidated formula, in essence, we divide the input signal with its standard deviation so that the result has unit standard deviation which allows for a more robustcomparison.

$$(\mathbf{x} - \mu_x)/\sigma_x$$

The comparison between the two images is performed on the basis of these 3 features.

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

The SSIM method is clearly more involved than the MSE method, but the gist is that SSIM attempts to model the perceived change in the structural information of the image, whereas MSE is actually estimating the perceived errors. There is a subtle difference between the two, but the results are dramatic.

SSIM is 0.6803057275071278 mean for a total of 1510 different predictions

V. CONCLUSION AND FUTURE SCOPE

The results of the model have been generated and presented. The Generative Adversarial model was successfully able topredict the 20th from the previous 19 frames. The quality of the 20th image is also recent and the comparisons of actual and predicted frames have also been done. The model was, although successful in creating the 20th frame from the training as well as the testing set, but was not able to entirely achieve its goal of predicting all possible frames in the future.The results of the model start to deteriorate after the 3rd iteration and vanish around the 5th iteration of frame generation. This is due to the time-insensitive nature of the model. As Conv3D blocks are used in the model, hence the model is unable to understand the flow of time within the frames. Also, because of its extreme noise sensitivity because of that, even a slight change (noise) in a frame causes the future frames to be even noisier which ultimately leads to this noise magnification in future frames generated from previous predicted frames.

The model is very good in predicting the immediate next future frame in a series. Hence even with a few modifications, itcan be used to predict frames not that far in the future. Which can be utilized in fast-paced machines that do not requiremuch foresight into the future. The results can be improved even more if LSTM blocks are used instead of Conv3D blockswhich could give the model an insight into understanding the motion of time and correctly predict better frames. This modification was not possible right now due to the extensive resources required to run that model with LSTM blocks. But by reducing the size of the image along with reducing the number of previous frames that the model looks at, a considerably smaller version of the model can be created which could successfully execute the entire model, creating better results in a reasonable amount of time. Hence, in the future, these few modifications could greatly improve the results produced by this model.

VI. ACKNOWLEDGEMENT

I would specially like to thanks my worthy guide Ms. Sakshi Jha, who supervised me to complete this research paper. Hertechnical advice, ideas and constructive criticism contributed to the success of this paper. She suggested me many ideas and solved my puzzles. Her motivation and help have been of great inspiration to me.

REFERENCES

[1] Oprea, Sergiu & Martinez-Gonzalez, Pablo & Garcia-Garcia, Alberto & Castro Vargas, John Alejandro & Orts, Sergio & Rodríguez, José & Argyros, Antonis. (2020). “A Review on Deep Learning Techniques for Video Prediction”. IEEE Transactions on Pattern Analysis and Machine Intelligence. PP. 1-1. 10.1109/TPAMI.2020.3045007.

[2] J. Hawkins and S. Blakeslee, On Intelligence. Times Books, 2004

[3] R. Baker, M. Dexter, T. E. Hardwicke, A. Goldstone, and Z. Kourtzi, “Learning to predict: Exposure to temporal sequences facilitates prediction of future events,” Vision Research, vol. 99, 2014.

[4] H. E. M. den Ouden, P. Kok, and F. P. de Lange, “How prediction errors shape perception, attention, and motivation,” in Front. Psychology, 2012

[5] Berner, Christopher & Brockman, Greg & Chan, Brooke & Cheung, Vicki & Dębiak, Przemysław & Dennison, Christy & Farhi, David & Fischer, Quirin & Hashme, Shariq & Hesse, Chris & Józefowicz, Rafal & Gray, Scott & Olsson, Catherine & Pachocki, Jakub & Petrov, Michael & Pinto, Henrique & Raiman, Jonathan & Salimans, Tim & Schlatter, Jeremy & Zhang, Susan. (2019). “Dota 2 with Large Scale Deep Reinforcement Learning”.

[6] I. Sutskever, G. E. Hinton, and G. W. Taylor, “The recurrent temporal restricted boltzmann machine,” in NIPS, 2008.

[7] N. Srivastava, E. Mansimov, and R. Salakhutdinov, “Unsupervised Learning of Video Representations using LSTMs,” in ICML, 2015.

[8] F. Ebert, C. Finn, A. X. Lee, and S. Levine, “Self-supervised visual planning with temporal skip connections,” in CoRL, ser. Proceedings of Machine Learning Research, vol. 78, 2017