ML 21/22-30 Test and Investigation of Video Learning Project

Shafait Azam (1345243)
shafait.azam@stud.fra-uas.de

Nayeem Ahmed (1344260)
nayeem.ahmed@stud.fra-uas.de

Abstract*—Hierarchical temporal memory is a machine learning model which is built upon the concept of pattern recognition. Unlike any other machine learning model, it works on the Sparse Distributed Representation to learn and recognize a specific pattern. The purpose of this paper is to analyze and test the existing video learning project by passing new videos to the HTM model, changing several parameters of the HTM configuration, manipulating frame size of the videos. Later, detail test and analysis have been conducted upon the migrated version of the video learning project.

***Keywords*—HTM Model, Video Sets, Frames, HTM parameters, Training Accuracy.**

I. Introduction

The purpose of the Hierarchical Temporal Memory (HTM) is to mimic the actual functionality of the neocortex which is situated in the human brain. It is a virtual model of neocortex capable of learning sequences of patterns and make predictions after learning. Human brain can process vast amount of information which are captured from environment by our sensory systems. To create semantical correlation among this raw sensory input our cortical neurons form synaptic connections with other neurons. For a specific input patterns, specific numbers of neuron participate in the learning mechanism. HTM follows kind of same structure. Numenta which is a private company developed HTM technology and made it available to researchers to develop this platform [1]. HTM is different from traditional deep neural network architectures. In HTM feature pooling depends on the temporal analysis of the pattern sequences but in traditional Convolutional Neural Network pooling is done by simple operators such as max, min or average. HTM utilizes time continuity for unsupervised learning of invariant representations which does not depend on the static and dynamic nature of the input patterns [1]. There are some important properties which can be exploited by HTM. The usage of time as a supervisor can solve the minor intra-class variations of patterns which can cause different spatial representations (in terms of pixel intensities, frame size in case of video learning). HTM is useful in this case which can claim that two representations originate from the same object because it exploits time continuity. The top down and bottom-up information flow provides HTM to dissimilate the situation while processing unambiguous patterns [1]. As it has been mentioned earlier that HTM can exploits time continuity, so it can be said that it is a system which can be trained on a sequence of events that vary over time [2]. These can be achieved by two core units know as spatial pooler (SP) and temporal memory (TM). In temporal memory the cortical learning algorithm (CLA) is performed. The SP is responsible for transforming the input data into sparse distributed representation (SDR) with fixed sparsity and the TM learns to recognize patterns from those SDRs and makes predictions [2].

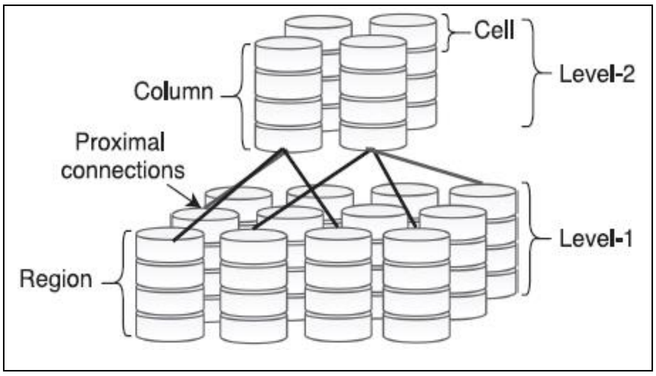

Fig. 1. Figure depicts the two HTM regions arranched in hirarchical order. Each region consists of vertical colums with stacked cells [2]

Here, Level -1 represents the SP which is consists of identical processing units that can create unique SDR from raw data. TM learns the patterns from those SDRs generated by SP and makes predictions from it. This neuromorphic architecture can serve as a core block for the HTM model [2]. The main contribution of this paper is to analyze and test an application which was built upon HTM. The Original video learning project was created but there were some lacking in the documentations and all the parameters had not been tested. Our project goal was to check all those parameters, analyze the correlations among them, check the training accuracy, create new videos and test the model with them.

II. Methodology

The existing video learning project was built to train videos and predicted sequences of frames from a given input frame which means that the model tried to recreate a video with proceeding frames. Small size videos like moving circle, triangle and rectangles were created to train and test the project. Initially, testing has been performed in the older version of video learning project [3]. After that, detailed analysis and testing with several parameters were conducted on the migrated version of the video learning project [4]. For the testing purpose pentagon, star and hexagon video set have been created by keeping the image gap duration between 0.2s to 0.4s. For creating this new video set online video maker named clideo has been used.

```
int frameWidth = 20;

int frameHeight = 20;

double frameRate = 12;
int maxCycles = 20;
```

Fig. 2. Figure depicts the parameters which were manipulated during testing the existing Video Learning Project [3].

The existing Video Learning project has two functions named Run1() and Run2(). Run2() is the sequence learning where sequences of frames from the videos passed to the HTM model. These both functions have been tested on the basis of the parameters described in the figure 2. Intially, the existing model has been tested with existing three video set (circle, rectangle and triangle). Afterwards, newly generated video sets named pentagon, star and hexagon have been gradually introduced to the model. The main purpose of conducting these experiments was to figure out how the model behaves when the data set is larger and analyse what are the consequences of changing the parameters described in figure 2. After all these experiments training accuracy of the model for different videoset has been documented. To test the prediction phase, a random frame from the converted video set folder has been chosen to check if the model can predict the next sequences of frames after passing that random frame as an input frame.

Later on, the migrated version of the video learning project has been tested with same videosets (circle, ractangle, triangle, pentagon, star and hexagon). But this time only Run2() which is the multi-sequence learning has been tested. By keeping the paramters described in the figure 2 stagnant, new parameters have been manupulated in this case in order to analyse the impact of those parameters.

```
"ColumnDimensions": [
    4096
  ],

 "CellsPerColumn": 80,
```

Fig. 3. Figure depicts the parameters which were manipulated during testing the migrated Video Learning Project [3].

The migrated version of the video learning project focuses mainly on the sequence learning. Experiments were conducted on this project with various video sets in order to measure the accuracy and also the runtime for each of those experiments. The main purpose of conducting such expermients in the migrated version of video learning was to slove the never ending newborn cycles generation phase.

III. Test Results

All the experiments that have been conducted for existing and migrated version of the video learning project; same video sets were used. The whole test process was divided into two parts.

A. *Experimentation with the existing project:*

The existing video learning project is divided into two functions named Run1() and Run2() [3]. Each function has been tested separately with six video sets named Circle, Rectangle, Triangle, Pentagon, Star and Hexagon.

1) *Run1() Experiment:*

Several test cases have been generated for the Run1() experimentations.

a) The existing Video Learning project was trained with three small video data set know as Circle, Triangle and Rectangle. Parameters were specified in the code and was used to run the Run1() function. By keeping all the parameters stagnant the HTM model has been trained with the SP + TM and documented the accuracy after 10 cycles.

TABLE I. RESULT GENERATED AFTER EXPERIMENT.

Video set	Training Accuracy
Circle	**45.714285714285715%**
Rectangle	**25.71428571428571%**
Triangle	**88.57142857142857%**

b) Afterwards we have changed the parameter ***maxCycles = 15*** *and kept every other parameter same. The following result was generated after 15 cycles.*

TABLE II. RESULT GENERATED AFTER EXPERIMENT.

Video set	Training Accuracy
Circle	**60%**
Rectangle	**37.142857142857146%**
Triangle	**82.85714285714286%**

c) It seems that the average accuracy after the training has been increased and the Triangle accuracy is dropped but the other two data set accuracy increased. To really measure the impact of ***maxCycles*** *parameter we have increased its number to* ***20*** *and calculated the following result.*

TABLE III. RESULT GENERATED AFTER EXPERIMENT.

Video set	Training Accuracy
Circle	**65.71428571428571%**
Rectangle	**42.857142857142854%**
Triangle	**100%**

d) Now we have increased the number of maxCycles parameter to 25 and generated the following result.

TABLE IV. RESULT GENERATED AFTER EXPERIMENT.

Video set	Training Accuracy
Circle	**45.714285714285715%**
Rectangle	**37.142857142857146%**
Triangle	**88.57142857142857%**

e) Now we have generated a new video dataset known as ***Pentagon****. Parameters were specified in the code and was used to run the* ***Run1()*** *function. By keeping all the parameters stagnant (HTM config) we have trained the HTM model with the SP + TM and documented the accuracy after* ***20*** *cycles. We have used 4 datasets in this experiment (Circle, Rectangle, Triangle and Pentagon).*

TABLE V. RESULT GENERATED AFTER EXPERIMENT

Video set	Training Accuracy
Circle	**45.714285714285715%**
Rectangle	**31.428571428571427%**
Triangle	**85.71428571428571%**
Pentagon	**67.1875%**

f) As we have trained the model with larger dataset (4 videos), we have increased the maxCycles parameter gradually to check the impact of this parameter on the average accuracy. So, we have changed the maxCycles parameter to 40 cycles and ran the experiment again. The following results were generated after 40 cycles. Here the frameWeight and frameHeight parameters are kept unchanged which is 18.

TABLE VI. RESULT GENERATED AFTER EXPERIMENT

Video set	Training Accuracy
Circle	**60%**
Rectangle	**42.857142857142854%**
Triangle	**100%**
Pentagon	**71.875%**

This results significantly show that if we increase the maxCycles parameter for larger data set than the average training accuracy increases. To validate our hypothesis, we have increased the specified parameters until the average accuracy reaches its saturation.

g) Afterwards we have trained the model with maxCycles = 70. We have noticed that if we train the model with 4 dataset it shows better training accuracy when the maxCycles is 40. The following results were generated after 70 cycles.

TABLE VII. RESULT GENERATED AFTER EXPERIMENT

Video set	Training Accuracy
Circle	**51.42857142857142%**
Rectangle	**45.714285714285715%**
Triangle	**67.1875%**
Pentagon	**88.57142857142857%**

h) Then we trained our model with ***40*** *maxCycles and we have change the parameters* ***frameWidth*** *and* ***frameHeight*** *to* ***20****. The following results were generated after 40 cycles.*

TABLE VIII. RESULT GENERATED AFTER EXPERIMENT

Video set	Training Accuracy
Circle	**57.14285714285714%**
Rectangle	**45.714285714285715%**
Triangle	**88.57142857142857%**
Pentagon	**68.75%**

i) Then we trained our model with ***40*** *maxCycles and we have change the parameters* ***frameWidth*** *and* ***frameHeight*** *to* ***22****. The following results were generated after 40 cycles.*

TABLE IX. RESULT GENERATED AFTER EXPERIMENT

Video set	Training Accuracy
Circle	**51.42857142857142%**
Rectangle	**34.285714285714285%**
Triangle	**85.71428571428571%**
Pentagon	**71.875%**

j) Then we trained our model with ***40*** *maxCycles and we have change the parameters* ***frameWidth*** *and* ***frameHeight*** *to* ***24****. The following results were generated after 40 cycles.*

TABLE X. RESULT GENERATED AFTER EXPERIMENT

Video set	Training Accuracy
Circle	**57.14285714285714%**
Rectangle	**42.857142857142854%**
Triangle	**85.71428571428571%**
Pentagon	**65.625%**

k) Now we have generated a new video dataset known as ***Hexagon****. Parameters were specified in the code and was used to run the* ***Run1()*** *function. By keeping all the parameters stagnant (HTM config) we have trained the HTM model with the SP + TM and documented the accuracy after* ***20*** *cycles. We have used 4 datasets in this experiment (Circle, Rectangle, Triangle and Hexagon).*

TABLE XI. RESULT GENERATED AFTER EXPERIMENT

Video set	Training Accuracy
Circle	**54.285714285714285%**
Rectangle	**37.142857142857146%**
Triangle	**91.42857142857143%**
Hexagon	**37.142857142857146%**

l) As we have trained the model with larger dataset (4 videos), we have increased the maxCycles parameter gradually to check the impact of this parameter on the average accuracy. So, we have changed the maxCycles parameter to 30 cycles and ran the experiment again. The following results were generated after 30 cycles. Here the frameWeight and frameHeight parameters are kept unchanged which is 18.

TABLE XII. RESULT GENERATED AFTER EXPERIMENT

Video set	Training Accuracy
Circle	**62.857142857142854%**
Rectangle	**48.57142857142857%**
Triangle	**94.28571428571428 %**
Hexagon	**67.56756756756756%**

This results significantly show that if we increase the maxCycles parameter for larger data set than the average training accuracy increases. To validate our hypothesis, we have increased the specified parameters until the average accuracy reaches its saturation.

m) Afterwards we have trained the model with maxCycles = 50. We have noticed that if we train the model with 4 dataset it shows better training accuracy when the maxCycles is 30. The following results were generated after 50 cycles.

TABLE XIII. RESULT GENERATED AFTER EXPERIMENT

Video set	Training Accuracy
Circle	**65.71428571428571 %**
Rectangle	**40 %**
Triangle	**97.14285714285714 %**
Hexagon	**59.45945945945946 %**

*n) Then we trained our model with **30** maxCycles and we have change the parameters **frameWidth** and **frameHeight** to **20**. The following results were generated after 30 cycles.*

TABLE XIV. RESULT GENERATED AFTER EXPERIMENT

Video set	Training Accuracy
Circle	**65.71428571428571 %**
Rectangle	**54.285714285714285 %**
Triangle	**100 %**
Hexagon	**66.21621621621621 %**

*o) Then we trained our model with **30** maxCycles and we have change the parameters **frameWidth** and **frameHeight** to **22**. The following results were generated after 40 cycles.*

TABLE XV. RESULT GENERATED AFTER EXPERIMENT

Video set	Training Accuracy
Circle	**68.57142857142857 %**
Rectangle	**37.142857142857146 %**
Triangle	**85.71428571428571 %**
Hexagon	**67.56756756756756 %**

*p) Then we trained our model with **30** maxCycles and we have change the parameters **frameWidth** and **frameHeight** to **24**. The following results were generated after 40 cycles.*

TABLE XVI. RESULT GENERATED AFTER EXPERIMENT

Video set	Training Accuracy
Circle	**54.285714285714285 %**
Rectangle	**45.714285714285715 %**
Triangle	**88.57142857142857 %**
Hexagon	**62.16216216216216 %**

*q) we have generated a new video dataset known as **Star**. Parameters were specified in the code and was used to run the **Run1()** function. By keeping all the parameters stagnant (HTM config) we have trained the HTM model with the SP + TM and **20** maxcycles. We have used **5 datasets** in this experiment **(Circle, Rectangle, Triangle, Pentagon and Star).** we changed the **frameHeight** and **frameWeight** parameters both to **20.** We have noticed one thing that when we increased the size of frame height and weight, the runtime of the whole process significantly decreased.*

TABLE XVII. RESULT GENERATED AFTER EXPERIMENT

Video set	Training Accuracy
Circle	**62.857142857142854%**
Rectangle	**45.714285714285715%**
Triangle	**88.57142857142857%**
Pentagon	**64.0625%**
Star	**63.74999999999999%**

*r) As we have trained the model with larger dataset **(5 videos)**, we have increased the maxCycles parameter gradually to check the impact of this parameter on the average accuracy. So, we have changed the maxCycles parameter to 40 cycles and ran the experiment again. The following results were generated after **40** cycles. Here the **frameWeight** and **frameHeight** parameters are kept unchanged which is **20**.*

TABLE XVIII. RESULT GENERATED AFTER EXPERIMENT

Video set	Training Accuracy
Circle	**48.57142857142857%**
Rectangle	**51.42857142857142%**
Triangle	**85.71428571428571%**
Pentagon	**71.875%**
Star	**58.75%**

s) Afterwards we have trained the model with maxCycles = 70. We have noticed that if we train the model with 5 dataset it shows better training accuracy when the maxCycles is 20. The following results were generated after 70 cycles.

TABLE XIX. RESULT GENERATED AFTER EXPERIMENT

Video set	Training Accuracy
Circle	**60%**
Rectangle	**45.714285714285715%**
Triangle	**85.71428571428571%**
Pentagon	**65.625%**
Star	**60%**

*t) Then we trained our model with **20** maxCycles and we have change the parameters **frameWidth** and **frameHeight** to **22**. The following results were generated after 20 cycles.*

TABLE XX. RESULT GENERATED AFTER EXPERIMENT

Video set	Training Accuracy
Circle	**54.285714285714285%**
Rectangle	**31.428571428571427%**
Triangle	**77.14285714285715%**
Pentagon	**62.5%**
Star	**70%**

*u) Then we trained our model with **20** maxCycles and we have change the parameters **frameWidth** and **frameHeight** to **24**. The following results were generated after 20 cycles.*

TABLE XXI. RESULT GENERATED AFTER EXPERIMENT

Video set	Training Accuracy
Circle	**54.285714285714285%**
Rectangle	**31.428571428571427%**
Triangle	**80%**
Pentagon	**59.375%**
Star	**70%**

2) Run2() Experiment:

Several test cases have been generated for the Run2() experimentations.

*a) The existing Video Learning project was trained with three small video data set know as Circle, Triangle and Rectangle. Parameters were specified in the code and was used to run the **Run2()** function. By keeping all the parameters stagnant we have trained the HTM model with the SP + TM and documented the accuracy after 1000 cycles. After the stable pattern was reached after each video set being trained, we documented the following result.*

TABLE XXII. RESULT GENERATED AFTER EXPERIMENT

Video Set	Cycle	Matches	Accuracy
Circle	72	25/30	**86.206%**
Rectangle	64	26/30	**89.655%**
Triangle	57	29/30	**100%**

*b) Afterwards we have changed the parameter **maxCycles = 1200** and kept every other parameter same.*

TABLE XXIII. RESULT GENERATED AFTER EXPERIMENT

Video set	Cycle	Matches	Accuracy
Circle	60	27/30	**93.103%**
Rectangle	64	24/30	**82.758%**
Triangle	57	28/30	**96.551%**

c) By keeping all the parameters stagnant we have trained the HTM model with the SP + TM (using 4 Videosets) and documented the accuracy after 1000 cycles. After the stable pattern was reached after each video set being trained, we documented the following result.

TABLE XXIV. RESULT GENERATED AFTER EXPERIMENT

Video set	Cycle	Matches	Accuracy
Circle	1000	18/30	**62.068%**
Rectangle	89	22/30	**79.028%**
Triangle	57	27/30	**93.103%**
Pentagon	1000	35/45	**66.037%**

*d) Afterwards we have changed the parameter **maxCycles = 1200** and kept every other parameter same.*

TABLE XXV. RESULT GENERATED AFTER EXPERIMENT

Video set	Cycle	Matches	Accuracy
Circle	1200	23/30	**79.310%**
Rectangle	163	24/30	**82.758%**
Triangle	57	27/30	**93.103%**
Pentagon	1000	35/45	**66.037%**

*e) So, we have seen that when we have increased the maxCycle parameter to **1200** the average training accuracy of the model is optimum for the given 4 videosets. Now we have manipulated the frameWeight and frameHeight parameters. We set these parameters to **20** and generated the following results.*

TABLE XXVI. RESULT GENERATED AFTER EXPERIMENT

Video set	Cycle	Matches	Accuracy
Circle	1200	19/30	**65.517%**
Rectangle	1200	20/30	**68.965%**
Triangle	58	25/30	**86.206%**
Pentagon	1200	33/45	**62.264%**

*f) Again, we have increased the frameWeight and frameHeight parameters to **22** and generated the following result.*

TABLE XXVII. RESULT GENERATED AFTER EXPERIMENT

Video set	Cycle	Matches	Accuracy
Circle	1200	23/30	**79.310%**
Rectangle	115	26/30	**89.655%**
Triangle	57	27/30	**93.103%**
Pentagon	1200	36/45	**70.264%**

B. Experimentation with the migrated project:

After the completion of testing with the existing video learning project, migrated version of the project has been analyzed and tested [4]. Same video sets with slightly different image gap duration were used in this purpose. Only Run2() function has been tested in the migrated version.

1) Run2() Experiment:

Several test cases have been generated for the Run2() experimentations.

*a) By keeping maxCycles, frameHeight, frameWidth and GetPredictedInputValues (3 for all cases) unchanged, we have manipulated ColumnDimensions and CellsPerColumn parameters in the htmConfig.json [4]. We have trained the model with our newly created pentagon video set and other three video sets (circle, rectangle and triangle). We have got the following results (training accuracy) when the ColumnDimensions = **2048** and CellsPerColumn = **40**. [We have used significantly larger video sets. That's, why we have started from higher column dimensions].*

TABLE XXVIII. RESULT GENERATED AFTER EXPERIMENT

Video set	Saturation reached after cycle	Accuracy	Total Runtime of the process
Circle	10	**100%**	**8 hrs. 14 minutes**.
Pentagon	56	**77.272%**	**8 hrs. 14 minutes**.
Rectangle	751	**100%**	**8 hrs. 14 minutes**.
Triangle	110	**100%**	**8 hrs. 14 minutes**.

*b) We have got the following results (training accuracy) when the ColumnDimensions = **2048** and CellsPerColumn = **80**.*

TABLE XXIX. RESULT GENERATED AFTER EXPERIMENT

Video set	Saturation reached after cycle	Accuracy	Total Runtime of the process
Circle	9	**100%**	**9 hrs. 47 minutes**.
Pentagon	188	**77.272%**	**9 hrs. 47 minutes**.
Rectangle	970	**80%**	**9 hrs. 47 minutes**.
Triangle	11	**100%**	**9 hrs. 47 minutes**.

*c) We have got the following results (training accuracy) when the ColumnDimensions = **2048** and CellsPerColumn = **160**.*

TABLE XXX. RESULT GENERATED AFTER EXPERIMENT

Video set	Saturation reached after cycle	Accuracy	Total Runtime of the process
Circle	8	**100%**	**8 hrs. 14 minutes**.
Pentagon	119	**77.2723%**	**8 hrs. 14 minutes**.
Rectangle	921	**91.428%**	**8 hrs. 14 minutes**.
Triangle	10	**100%**	**8 hrs. 14 minutes**.

*d) By keeping maxCycles, frameHeight, frameWidth and GetPredictedInputValues (3 for all cases) unchanged, we have manipulated ColumnDimensions and CellsPerColumn parameters in the htmConfig.json [4]. We have trained the model with our newly created Star video set and other four video sets (circle, rectangle, triangle and pentagon). We have got the following results (training accuracy) when the ColumnDimensions = **2048** and CellsPerColumn = **40**. [We have used significantly larger video sets (5 videos). That's, why we have started from higher column dimensions].*

TABLE XXXI. RESULT GENERATED AFTER EXPERIMENT

Video set	Saturation reached after cycle	Accuracy	Total Runtime of the process
Circle	9	**100%**	**13 hrs. 19 minutes**.
Pentagon	83	**77.272%**	**13 hrs. 19 minutes**.
Rectangle	831	**100%**	**13 hrs. 19 minutes**.
Triangle	563	**91.667%**	**13 hrs. 19 minutes**.
Star	112	**85.714%**	**13 hrs. 19 minutes**

*e) We have got the following results (training accuracy) when the ColumnDimensions = **2048** and CellsPerColumn = **80**.*

TABLE XXXII. RESULT GENERATED AFTER EXPERIMENT

Video set	Saturation reached after cycle	Accuracy	Total Runtime of the process
Circle	10	**100%**	**15 hrs. 26 minutes**.
Pentagon	122	**77.272%**	**15 hrs. 26**

			minutes.
Rectangle	700	**97.1428%**	**15 hrs. 26 minutes**.
Triangle	665	**85.714%**	**15 hrs. 26 minutes**.
Star	130	**79.1667%**	**15 hrs. 26 minutes**

*f) We have got the following results (training accuracy) when the ColumnDimensions = **2048** and CellsPerColumn = **160**.*

TABLE XXXIII. RESULT GENERATED AFTER EXPERIMENT

Video set	Saturation reached after cycle	Accuracy	Total Runtime of the process
Circle	10	**100%**	**18 hrs. 21 minutes**.
Pentagon	100	**77.273%**	**18 hrs. 21 minutes**.
Rectangle	767	**97.1428%**	**18 hrs. 21 minutes**.
Triangle	666	**91.667%**	**18 hrs. 21 minutes**.
Star	54	**91.4285%**	**18 hrs. 21 minutes**

*g) We have got the following results (training accuracy) when the ColumnDimensions = **4096** and CellsPerColumn = **80**.*

TABLE XXXIV. RESULT GENERATED AFTER EXPERIMENT

Video set	Saturation reached after cycle	Accuracy	Total Runtime of the process
Circle	10	**100%**	**27 hrs. 7 minutes**.
Pentagon	65	**68.181%**	**27 hrs. 7 minutes**.
Rectangle	805	**100%**	**27 hrs. 7 minutes**.
Triangle	665	**88.571%**	**27 hrs. 7 minutes**.
Star	99	**83.333%**	**27 hrs. 7 minutes**

IV. Discussion

As we have analyzed and tested both the existing and migrated version of the project, we found the pros and cons of both versions. Previous version that we have tested have lower training accuracy per video set than the migrated version. Migrated version is efficient in terms of accuracy, but it takes a lot of time when we tried to run the HTM model with significantly larger data sets. Both Run1() and Run2() functions of the pervious version of the project runs faster. We have generated three new video sets to test the model. While experimenting with the migrated version of project, we came across an important issue which is in the case of larger video where we have frames which are identical to each other can create problem in the training phase. When the model reaches the newborn cycle phase it kept generating the newborn cycles because it can not figure out the stopping point of learning. Then we have made slight changes in our generated videos by reducing the image gap between to frames to 0.2s which solved this problem. From this we have drawn the conclusion that, the model is still not entirely ready to process larger videos. This can be improved in the future work. Also, the prediction phase of the project did not reach our expectations. Because when we passed a frame, the model should return the proceeding frames and try to create a video from those frames. It did not work all the time. Again, prediction accuracy can also be improved in the future work. Using different codec can be a way to improve the prediction phase.

We have seen that our pentagon video set reaches 72.273% training accuracy and star video set reaches 91.4285% when we tested the migrated version. The migrated version performs well with our new videos in the training phase, but the run time is high. This project can still be improved. We thought of adding another layer can solve the prediction problem because a single layer is working with relatively larger data set. For this the overlap connection must be checked. Another solution can be done by changing the HTM Configuration. In our experiments, most of the parameters of HTM model were unchanged. The impact of those parameters in the video learning project is still unknown. Those parameters can be monitored with breaking points and analyzed each of them carefully. Our findings proved that the migrated version is better in terms of accuracy then the previous version. But the project is still not ready to process larger video sets with similar frames next to each other as it could not identify them in the training phase.

V. Reference

[1] Maltoni, D. (2011). Pattern recognition by hierarchical temporal memory. *Available at SSRN 3076121*.

[2] A. M. Zyarah and D. Kudithipudi, "Neuromorphic Architecture for the Hierarchical Temporal Memory," in *IEEE Transactions on Emerging Topics in Computational Intelligence*, vol. 3, no. 1, pp. 4-14, Feb. 2019, doi: 10.1109/TETCI.2018.2850314.

[3]https://github.com/ddobric/neocortexapi/tree/SequenceLearning_ToanTruong

[4]https://github.com/ddobric/neocortexapi-videolearning