# 25. Multi-Sequence/Image Learning Project (WS21/22) Project - Team Noobies

Students:

1. Harish Palanivel, 1392283 ([harish.palanivel@stud.fra-uas.de](mailto:harish.palanivel@stud.fra-uas.de))
2. Gaurav Honnavara Manjunath, 1384178 ([gaurav.honnavaramanjunath@stud.fra-uas.de](mailto:gaurav.honnavaramanjunath@stud.fra-uas.de))
3. Athkar Praveen Prajwal, 1394663 ([praveen.athkar@stud.fra-uas.de](mailto:praveen.athkar@stud.fra-uas.de))

## 1. Motivation:

In the following experiment, we introduce different types of sequences into the HTM and Use HTM Image Encoder for Image Sequence Learning.

Experiment 1 – Sequence Learning with Numbers.

Experiment 2 - Sequence Learning with Alphabets.

Experiment 3 - Anti Cancer Peptides Sequence Classification.

Experiment 4 – Implementing Multi-Image Sequence Learning Using HTM Image Encoder.

## 2. Overview:

This project references

<ul>

<li>Sequence Learning sample, see [SequenceLearning.cs](https://github.com/ddobric/neocortexapi/tree/master/source/Samples/NeoCortexApiSample). </li>

<li>Video Learning sample, see [VideoLearning.cs] ([https://github.com/ddobric/neocortexapi/blob/SequenceLearning\_ToanTruong/Project12\_HTMCLAVideoLearning/HTMVideoLearning/HTMVideoLearning/VideoLearning.cs)</li](https://github.com/ddobric/neocortexapi/blob/SequenceLearning_ToanTruong/Project12_HTMCLAVideoLearning/HTMVideoLearning/HTMVideoLearning/VideoLearning.cs)%3c/li)>

<li>https://github.com/prajwalpraveen97/neocortexapi/blob/prajwalpraveen97\_ML/source/ImageEncoder/ImageEncoder.cs</li>

</ul>

The learning process includes:

1. reading sequences.

2. encoding the data using encoders.

3. Spatial Pooler Learning with Homeostatic Plasticity Controller until reaching a stable state.

4. Learning with Spatial pooler and Temporal memory, conditional exit.

5. Interactive testing section, output classification/prediction from input data.

6. Implementing Multi-Image Sequence Learning Using Image Encoder.

## 3. Data Format:

Experiment 1 - Sequence Learning with Numbers

DataFormat - [Number Sequence] -> [Sequence Class]

Sequences - Multi Sequence - Here Alphabetic Sequence is considered a sequence of characters.

Example Datarow - 0.0, 1.0, 0.0, 2.0, 3.0, 4.0, 5.0, 6.0, 5.0, 4.0, 3.0, 7.0, 1.0, 9.0, 12.0, 11.0, 12.0, 13.0, 14.0, 11.0, 12.0, 14.0, 5.0, 7.0, 6.0, 9.0, 3.0, 4.0, 3.0, 4.0, 3.0, 4.0

Experiment 2 - Sequence Learning with Alphabets

DataFormat - [Alphabetic Sequence] -> [Sequence Class]

Sequences - Multi Sequence - Here Alphabetic Sequence is considered sequence of characters.

Example Datarow - AIADISAASIFIIISIFF

Experiment 3 - Anti Cancer Peptides Sequence Classification

DataFormat - [Alphabetic Sequence] -> [Sequence Class]

Sequences - Multi Sequence - Here Alphabetic Sequence is considered sequence of characters.

Example Datarow - AAWKWAWAKKWAKAKKWAKAA, mod. Active

Experiment 4 – Implementing Multi-Image Sequence Learning Using HTMImageEncoder.

<TBD>

## 4. Learning Process:

<ul>

<li>For all the experiments same configuration has been used.</li>

<li>For Experiment 4, We have used Image Encoder and Optimized the code accordingly. </li>

<li>Current HTM Configuration:</li>

</ul>

```csharp

HtmConfig cfg = new HtmConfig(new int[] { inputBits }, new int[] { numColumns })

{

Random = new ThreadSafeRandom(42),

CellsPerColumn = 32, // Config 2 => 25

GlobalInhibition = true,

LocalAreaDensity = -1,

NumActiveColumnsPerInhArea = 0.02 \* numColumns,

PotentialRadius = 65,

InhibitionRadius = 15,

MaxBoost = 10.0,

DutyCyclePeriod = 25,

MinPctOverlapDutyCycles = 0.75,

MaxSynapsesPerSegment = 128,

ActivationThreshold = 15,

ConnectedPermanence = 0.5,

// Learning is slower than forgetting in this case.

PermanenceDecrement = 0.25,

PermanenceIncrement = 0.15,

// Used by punishing of segments.

PredictedSegmentDecrement = 0.1

};

```

### 1. SP Learning with HomeoStatic Plasticity Controller (HPA):

This first section of learning use Homeostatic Plasticity Controller:

```csharp

HomeostaticPlasticityController hpa = new HomeostaticPlasticityController(mem, Sequences.Count, (isStable, numPatterns, actColAvg, seenInputs) =>

{

if (isStable)

// Event should be fired when entering the stable state.

Console.WriteLine($"STABLE: Patterns: {numPatterns}, Inputs: {seenInputs}, iteration: {seenInputs / numPatterns}");

else

// Ideal SP should never enter unstable state after stable state.

Console.WriteLine($"INSTABLE: Patterns: {numPatterns}, Inputs: {seenInputs}, iteration: {seenInputs / numPatterns}");

// We are not learning in instable state.

learn = isInStableState = isStable;

// Clear all learned patterns in the classifier.

//cls.ClearState();

}, numOfCyclesToWaitOnChange: 30);

```

### 2. SP+TM Learning:

This is second phase of training, SP+TM are trained together after SP training

```csharp

foreach (var sequence in Sequences) // SEQUENCE LOOP

{

int SequencesMatchCount = 0; // NUMBER OF MATCHES

var tempLOGFILE = new Dictionary<int, string>();

int MatchesCount = 0;

double SaturatedAccuracyCount = 0;

for (int i = 0; i < maxCycles; i++) // MAXCYCLE LOOP

{

cycle++;

List<string> ElementWisePrediction = new List<string>();

List<string> ElementWiseClasses = new List<string>();

// ELEMENT IN SEQUENCE MATCHES COUNT

int ElementMatches = 0;

foreach (var Elements in sequence) // SEQUENCE DICTIONARY LOOP

{

// OBSERVATION LABEl

var observationLabel = Elements.Key;

// ELEMENT SDR LIST FOR A SINGLE SEQUENCE

var ElementSdr = Elements.Value;

List<Cell> actCells = new List<Cell>();

var lyrOut = new ComputeCycle();

lyrOut = layer1.Compute(ElementSdr, learn) as ComputeCycle;

Debug.WriteLine(string.Join(',', lyrOut.ActivColumnIndicies));

// Active Cells

actCells = (lyrOut.ActiveCells.Count == lyrOut.WinnerCells.Count) ? lyrOut.ActiveCells : lyrOut.WinnerCells;

if (!classVotingEnabled)

{

cls.Learn(observationLabel, actCells.ToArray());

if (lastPredictedValue == observationLabel && lastPredictedValue != "")

{

ElementMatches++;

Debug.WriteLine($"Match. Actual value: {observationLabel} - Predicted value: {lastPredictedValue}");

}

else

{

Debug.WriteLine($"Mismatch! Actual value: {observationLabel} - Predicted values: {lastPredictedValue}");

}

}

else

{

cls.Learn(HelperMethods.processLabel(observationLabel), actCells.ToArray());

}

```

## 5. Experiment Details:

In the following section experiments performed are explained in detail with training and testing phase results.

### 1. Anti-Cancer Peptide Sequence Classification:

#### Data Preparation and Processing:

<ul>

<li>Dataset are available at https://archive.ics.uci.edu/ml/datasets/Anticancer+peptides. </li>

<li>We are fetching data from Training File Directory using. </li>

<li> We are using elementwise prediction and later applying majority votes value as classification/label value. </li>

</ul>

```csharp

public static List<Dictionary<string, List<string>>> ReadCancerSequencesDataFromFile(string dataFilePath)

```

see,[HelperMethods.cs](https://github.com/prajwalpraveen97/neocortexapi/blob/prajwalpraveen97\_ML/MyProjectWork/SequenceLearningExperiments/SequenceLearningExperiments/HelperMethods.cs).

#### DataEncoding

we encode data using a scalar encoder for converting alphabet numeric values.

#### AlphabetEncoder

```csharp

Dictionary<string, object> settingsScalarEncoder\_Alphabets = new Dictionary<string, object>()

{

{ "W", 5},

{ "N", 31},

{ "Radius", -1.0},

{ "MinVal", (double)1},

{ "Periodic", true},

{ "Name", "scalar"},

{ "ClipInput", false},

{ "MaxVal", (double)27}

};

ScalarEncoder encoder\_Alphabets = new ScalarEncoder(settingsScalarEncoder\_Alphabets);

```

#### Example of data encoding:

<ul>

<li>Raw Data Row : AAWKWAWAKKWAKAKKWAKAA [Each alphabet will be encoded seprately]</li>

<li>Encoded Data Row : A 1:1:1:0:0:0:0:0:0:0:0:0:0:0:0:0:0:0:0:0:0:0:0:0:0:0:0:0:0:1:1</li>

<li>W 0:0:0:0:0:1:1:1:1:1:0:0:0:0:0:0:0:0:0:0:0:0:0:0:0:0:0:0:0:0:0</li>

<li>K 0:0:0:0:0:0:0:0:0:0:1:1:1:1:1:0:0:0:0:0:0:0:0:0:0:0:0:0:0:0:0</li>

<li>....</li>

</ul>

#### Testing Modes:

#### Version 1:

Explanation:

(To Be Modified as per our Input)

Input: FKVKFKVKVK, inactive - exp\_44

Learning: (F\_inactive - exp\_44)-(K\_inactive - exp\_44 )...-(K\_inactive - exp\_44)

Prediction: F-> k,... k-> v,...

<Output Image To Be Uploaded >

#### Version 2:

Explanation:

(To Be Modified as per our Input)

Input: FKVKFKVKVK, inactive - exp\_44

Learning: (FKVKFKVKVK\_inactive - exp\_44)

Note: We are using the whole sequence as a single element and passing it with multiple sequences in a .csv file

<Output Image To Be Uploaded>

## Similar Studies/Research used as References

[1] Continuous online sequence learning with an unsupervised neural network model.

Author: Yuwei Cui, Subutai Ahmad, Jeff Hawkins| Numenta Inc.

[2] On the performance of HTM predictions of Medical Streams in real-time.

Author: Noha O. El-Ganainy, Ilangkp Balasingham, Per Steinar Halvorsen, Leiv Arne Rosseland.

[3] Sequence memory for prediction, inference, and behaviour

Author: Jeff Hawkins, Dileep George, Jamie Niemasik | Numenta Inc.

[4] An integrated hierarchical temporal memory network for real-time continuous multi interval

prediction of data streams

Author: Jianhua Diao, Hyunsyug Kang.

[5] Stock Price Prediction Based on Morphological Similarity Clustering and Hierarchical Temporal

Memory

Author: XINGQI WANG, KAI YANG, TAILIAN LIU