|  |  |
| --- | --- |
| Information Technology Course  Module Software Engineering by  Damir Dobric / Andreas Pech |  |

Implementation of Scalar Encoder with Buckets in HTM

|  |  |  |
| --- | --- | --- |
| Aqib Javed | Haris Abbas Qureshi |  |
| [aqib.javed@stud.fra-uas.de](mailto:aqib.javed@stud.fra-uas.de) | [haris.abbas@stud.fra-uas.de](mailto:haris.abbas@stud.fra-uas.de) |  |

**Abstract—** Scalar Encoder is one of the encoding techniques and is a part of Hierarchical Temporal Memory (HTM). The Scalar Encoder's main purpose is to convert scalar values (such as numeric or floating-point values) into a distributed representation that can be used by HTM algorithms. It does this by transforming the input value into a binary array of bits, where each bit position corresponds to a different feature or dimension of the input value. The Scalar Encoder generates a sparse distributed representation, where only a small subset of bits in the array are active (i.e., set to 1) for any given input value. The active bits form a contiguous block of 1's, whose location within the array varies continuously depending on the input value.

***Keywords- HTM, neocortex, numeric, array, block, scalar encoder, bucket***

1. **INTRODUCTION**

To compare the Scalar Encoder in NeocortexApi with the Scalar Encoder developed by Numenta, we would need to examine the implementation details of both encoders and compare their performance on different types of input data. As for developing a method to validate the bucket index, we can perform the several steps : Determine the minVal, maxVal, n, and w values for the Scalar Encoder being used ,Generate a set of random input values within the range of minVal and maxVal ,Encode each input value using the Scalar Encoder and obtain the corresponding bit sequence ,Calculate the bucket index for each input value using the following formula: bucketIndex = sum of all on bits in the bit sequence ,Verify that the bucket index is within the range of 0 to n ,Repeat steps 2 to 5 for multiple input values and calculate the average bucket index ,Compare the average bucket index with the expected value based on the nand w parameters of the Scalar Encoder. The expected value can be calculated as: expectedBucketIndex = w \* (inputValue - minVal) / (maxVal - minVal).

If the average bucket index matches the expected value, then the method for calculating the bucket index is correct. However, if there is a significant difference between the average bucket index and the expected value, then there may be an issue with the implementation of the Scalar Encoder or the method for calculating the bucket index.

# METHODOLOGY

Encoder should create SDRs, no matter what it represents, that have a fixed number of bits ‘N’ and fixed number of active (1’s) bits ‘W’. What do you know which values are perfect for N and W?

We cannot be a big fraction of N, to preserve the properties that come from sparsity. But if W is too small then we lose the properties resulting from a distributed representation.

There are a number of specific aspects to consider when encoding the data:

1. Choosing appropriate values for N and W: N should be large enough to allow for a detailed representation of the input data, while W should be a small fraction of N to preserve sparsity. A common range for N is between 100 and 1000, while W is typically around 20-25% of N.

2. Preserving semantic relationships: The encoder should be designed to capture semantically related data, so that inputs with overlapping characteristics will generate overlapping SDRs with active bits in common.

3. Deterministic output: The encoder should always generate the same output SDR for the same input, to avoid redundancy in the learned sequence in HTM systems.

4. Fixed output dimensionality: The encoder's output should always have the same number of bits, regardless of the input, to enable comparisons and other operations.

5. Robustness to noise and subsampling: The encoder should include enough one-bits to accommodate noise and subsampling, with a general rule of thumb of at least 20-25 one-bits per SDR.

When we construct an encoder implementation, we first divide the range of values into buckets and then map the buckets into a collection of active cells.

The index i is calculated based on the input value, using the formula i = (input value - MinVal) \* (N - W) / Range.

The W bits starting from index i are set to active to represent the input value.

The remaining bits in the output are left unset.

This process is repeated for each input value, resulting in an encoded representation for the entire dataset. The resulting output will be a binary vector of length N, with W bits set to 1 in each segment that represents a particular input value. The encoder is deterministic, so the same input value will always result in the same binary vector representation. This allows for efficient comparison and processing of the encoded data in downstream algorithms such as HTM. The bucketing approach used in this encoder implementation helps to group similar input values into the same collection of active cells, making it easier to identify patterns and similarities in the data. The choice of MinVal, MaxVal, W, and N will depend on the specific characteristics of the input data, such as the range of values, resolution required, and the desired level of noise tolerance. It's important to strike a balance between having enough bits to represent the input values accurately and efficiently, while also ensuring that the resulting binary vector is sparse enough to avoid overlap and maintain distinguishability between different input.

1. **TEST CASES WITH RESULTS**

# 2. Test Case to encode Month of Year.

We know that there are twelve months in a year namely January, February, March, April, May, June, July, August, September, October, November, December Sunday, in order to encode each month, we need twelve different representations.

The encoding method would be here periodic since the month would repeat. Remember that there are four parameters to this encoding scheme: minimum value, maximum value, number of bits (N) and number of active bits (W).

1. This MinVal is 0 (January) and the MaxVal 12 (December).
2. The range is calculated with the formula MaxVal – MinVal = 11.
3. The number of bits that are set to encode a single value the ‘width’ of output signal ‘W’ used for representation is 3.
4. Total number of bits in the output ‘N’ used for representation is 14.
5. We are choosing the value of N=14 and W=3 to get the desired bucket output which shifts between January to December like shown below.

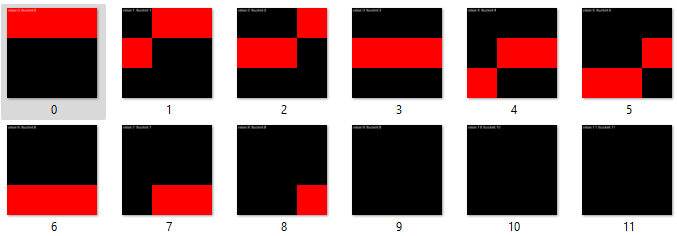


Fig.2.1. Expected Output of Months of Year

1. If we choose any other values for N and W for example N=10 and W=3 then it does not match the expected bucket output.

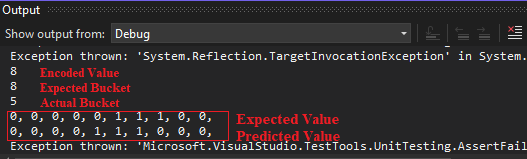


Fig.2.2. Expected Output of Months of Year

1. This example is periodic because the months of the year keep repeating for every 13th value, so resolution has to be calculated based on formula Range/N = 0.78.
2. The representation will be 14 bits with 3 consecutive active bits starting at the 4th bit as shown below

3 active bits



0000**111**0000000

4th bit

1. Once all the inputs are encoded, we call the Bitmap method to show the output in 2D Bitmap format. After setting all the parameter values run the program, the output images will be captured and saved in a folder it will show how the shifting is happening for every month of year.

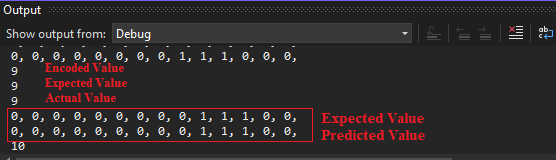


Fig.2.3. Expected Output of Months of Year

# 3. Test case to encode Bus Availability in a Station.

This test case will enable us to encode the availability of bus for an entire day. Assuming that the Buses will arrive every 60 mins. Firstly the 24 hours clock will be converted into minutes which will be equal to 1440 minutes a day.

1. Since it is a day clock in minutes it starts from 0 and end at 1440, hence the MinVal = 0 and MaxVal

= 1440

1. Computing the Range = MaxVal – MinVal which is equal to 1440.
2. Choosing the value of ‘W’ and ‘N’ in such way that there should be a shift for every 60 minutes in the output.
3. Total number of bits in the output ‘N’ used for representation is 11.
4. The number of bits that are set to encode a single value the ‘width’ of output signal ‘W’ used for representation is 11 is.
5. We are choosing the value of N=24 and W=11 to get the desired output.

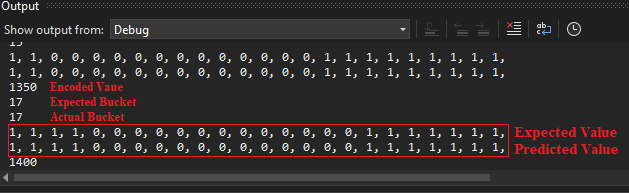


Fig.3.1. Output of Buses Availability

1. If we choose any other values for N and W for example N=16 and W=11 then it does not match the expected output.

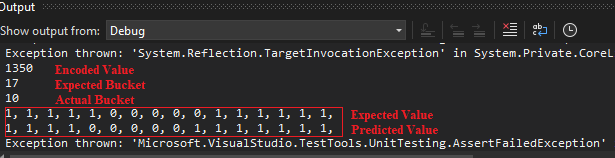


Fig.3.2. Output of Buses Availability

1. The time interval between adjacent Buses can be changed by altering the values of N and W for the known MinVal and MaxVal.

Once all the inputs are encoded, we call the Bitmap method to show the output in 2D Bitmap format.

After setting all the parameter values run the program, the output images will be captured and saved in a folder.

Bitmap method is executed in the code to produce these data in 2D Bitmap format.

In this availability of train test case, there is a shift after every 60 minutes which is in between 0 to 1440, As it is periodic most of the bit overlap in adjacent values as we can see in the below figure Fig.2.3(Output of Buses Availability) which is the snapshot of output of encoded availability of train test case.

# 

Fig.3.3. Output of Buses Availability

# 4. Test case to encode Ticket Number for Music Show.

Encoding the ticket number where people participate with unique ticket number in a Music concert. Let us say we have Premium, Golden, Silver and Classic tickets for Music concert. We must differentiate sections based on which category choosing by users.

Assuming the music concert have total 100 number of tickets available for crowd to participate in a show and also concert divide into four different categories, so according to assign ticket number people enter in a show. So, we must ensure that everyone has their unique ticket number so their will be no overlapping.

1. Tickets number range from 0 to 100, Hence the MinVal = 0 and MaxVal = 100
2. Computing the Range = MaxVal – MinVal which is equal to 100.
3. Hence the number of bits that are set to encode a single value the ‘width’ of output signal ‘W’ used for representation is 11.
4. Total number of bits in the output ‘N’ used for representation is 21.
5. We are choosing the value of N=21 and W=11 to get the desired output which is shown below:



Fig.4.1. Output of Ticket Number

1. If we choose any other values for N and W for example N=18 and W=3 then it does not match the expected bucket output.



Fig.4.2. Output of Ticket Number

Once all the inputs are encoded, we call the Bitmap method to show the output in 2D Bitmap format.

After setting all the parameter values run the program, the output images will be captured and saved in a folder.

Bitmap method is executed in the code to produce these data in 2D Bitmap format.

In this tickets number in Music concert test case, there is a several range of ticket numbers mentioned above and we can see the same in the below figure Fig.4.1(Output of Ticket Number for Music Show) which is the snapshot of output of encoded Ticket Number for Music Show case.



Fig.4.3. Output of Ticket Number

# Test Case to encode Age of Employees

**in Company**

Encoding the different category of people employees in Company according to their age. Let us say we have teenagers, adults, middle age and senior citizens employees in Company. We must differentiate employees based on this category choosing the bracket of age.

Assuming the people entering have ages in the range of 0 year to 59 years. We would like to encode differently example 0-18 years as one category and other category such as young adult range 19-39 years, middle age range 35-49, Senior age range 50+ years.

So, we are encoding different category age of people in different way.

1. Age of employees is in between 0 to 59 years, Hence the MinVal = 0 and MaxVal = 59.
2. Computing the Range = MaxVal – MinVal which is equal to 59.
3. Hence the number of bits that are set to encode a single value the ‘width’ of output signal ‘W’ used for representation is 3.
4. Total number of bits in the output ‘N’ used for representation is 7.
5. We are choosing the value of N=7 and W=3 to get the desired output which is shown below:



Fig.5.1. Output of Ticket Number

1. If we choose any other values for N and W for example N=6 and W=3 then it does not match the expected output which is shown below:



Fig.5.2. Output of Ticket Number

Once all the inputs are encoded, we call the Bitmap method to show the output in 2D Bitmap format.

After setting all the parameter values run the program, the output images will be captured and saved in a folder.

Bitmap method is executed in the code to produce these data in 2D Bitmap format.

In this Age of employees in Company test case, there is a shift within those categories mentioned above and we can see the same in the below figure Fig.9(Output of Age of employees in Company) which is the snapshot of output of encoded Age of employees in Company test case.



Fig.5.3. Output of Ticket Number

# Test Case to encode Temperature Ranges