*Implement Serialisation of HTMClassifier*

Amith Nair  
amith.nair@stud.fra-uas.de Aneeta Antony  
aneeta.antony@stud.fra-uas.de

Rohit Suresh  
rohit.suresh@stud.fra-uas.de

*Abstract*— Serialization is the act of transforming an object or data structure into a format that is simple to store, send, or exchange with other systems or applications. A text-based format, a binary stream, or a structured file format like JSON or XML can all be used for the serialized representation. This paper focuses on the implementation of serialization in Hierarchical Temporal Memory (HTM) classifier and demonstrates how the parameters of HTM classifier can be serialized. A machine learning method called the HTM classifier learns and anticipates patterns in time-series data using the neocortex's structural layout. In this project, Neocortex API focuses implementation of serialization Hierarchical Temporal Memory Classifier in C#/.NET Core. Serialization in the HTM classifier enables the classifier to preserve its state and structure as a file, enabling it to maintain its state over different runs or transfer its state to a different machine. It is helpful for picking up where training stopped off and incorporating the classifier into different programs or frameworks. We have carried out several tests by comparing the serialization and deserialization results in order to get the required outcome.

Keywords—: Classifier, Hierarchical Temporal Memory, Sparse Distributed Representation, neocortex.

# Introduction

A new paradigm in the study of artificial intelligence has emerged as a result of the idea of Hierarchical Temporal Memory (HTM) and Cortical Learning Algorithms (CLA) recently developing. A biomimetic model called HTM-CLA is based on Jeff Hawkins' memory-prediction hypothesis of brain activity. It is a technique for identifying and extrapolating the root causes of observable input patterns and sequences in order to construct an ever-more sophisticated model of the real world. The structure and operation of the human neocortex are the foundation of HTM-CLA. It lays the foundation for creating robots that do numerous cognitive activities at levels close to or higher than those of humans. It is utilized by the NuPIC, a Numenta-led open-source project. Fundamentally, HTM-CLA is a memory-based prediction system. The networks hold a substantial number of patterns and sequences and are trained on time-varying input. It is intrinsically time based and organized hierarchically. HTM-CLA acquires and saves the hierarchy of regions' structure and sequences of data in a special representation known as Sparse Distributed Representation (SDR).

This paper focuses on the application of serialization in an HTM classifier with the goal of outlining the procedure in great detail and emphasizing the advantages of this method. The purpose of serialization is to enable the trained model to be stored to a file or database for subsequent use in an HTM (Hierarchical Temporal Memory) classifier. A stream of bytes that may be saved on storage or sent over a network is created by serializing an object that is now in memory. Deserialization is the process of building an object from stored bytes in the opposite direction. When an HTM classifier has been trained using a set of data, its state may be saved by serializing the classifier. As a result, we can reuse the model instead of always having to train it from scratch. This is especially helpful in situations when the training procedure is computationally expensive or the training data is huge. Moreover, serialization makes it possible to share the model across many contexts or apps. For usage in production settings, a trained HTM classifier, for instance, might be serialized and disseminated to other systems. Overall, trained models become more portable and effective because to serialization, which offers a mechanism to preserve and reuse them.

# Methods

## A) HTM

The biological processes of the brain and its learning mechanism are the foundation of HTM. The findings are highly pertinent and demonstrate a small percentage of incorrect predictions over time. The goal of the theory and machine learning technology known as the Hierarchical Temporal Memory Cortical Learning Algorithm (HTM CLA) is to model the cortical algorithm of the neocortex.

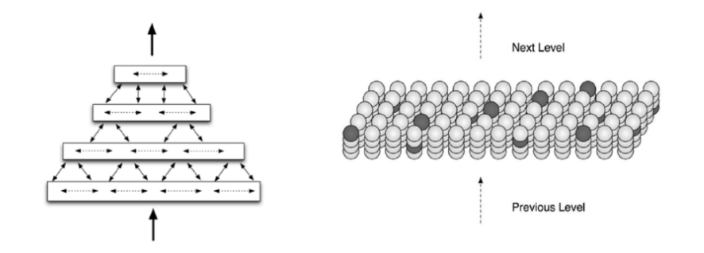


Fig.1 (a) Hierarchies in HTM-CLA (b) Cells organized in columns in a single region

The hierarchical structure of HTM-CLA is depicted in Figure 1. The cell in HTM-CLA is the fundamental unit of hierarchy. Columns are used to arrange the cells. These columns combine to create an area, which in turn, creates hierarchies. The cells can link to other cells at higher or lower levels that are located in the same region or regions. While columns indicate the semantics of the data using SDR representations, cells learn and store the temporal sequence of the data.

## B) SDR

To represent patterns in the incoming data, HTM systems employ sparse distributed representations (SDRs), which serve as its core data structure. SDRs are binary vectors with a disproportionately high proportion of zeros and a disproportionately low proportion of ones, with the ones being scattered randomly across the vector.

The classifier, one of the essential parts of an HTM system, is in charge of identifying and categorizing patterns in the input data. The classifier learns and recognizes patterns over time by combining spatial pooling with temporal memory. SDRs are generated from the input data using spatial pooling, and SDR sequences are stored and recognized using temporal memory.

SDRs are particularly well-suited for usage in the HTM classifier due to their mathematical characteristics. For instance, the classifier can effectively represent a large number of patterns using a relatively small number of bits because to the SDRs' sparsity. The distributed representation of patterns also enables the classifier to identify patterns in the presence of missing or truncated bits in the SDR.

Also, the classifier may use similarity-based categorization because to the mathematical characteristics of SDRs. The classifier can assess how similar two patterns are by comparing the overlap between their SDRs. As a result, even if new patterns are not precise matches to those it has already encountered, the classifier may still detect them.

## C) Classifier

An essential part of an HTM system, the classifier is in charge of identifying and categorizing patterns in the input data. It learns and recognizes patterns over time by combining temporal memory with spatial pooling. Assigning each input bit to one or more columns in a grid during spatial pooling results in the creation of SDRs from the input data. Sequences of SDRs are stored and recognized in temporal memory, which enables the classifier to identify patterns that develop over time SDRs are used to represent the visual aspects of the image to show how the HTM classifier performs well at identifying abnormalities in crowded settings.

## D) Serilization

An object or data structure is serialized when it is put into a format that makes it simple to store or send across a network. It is frequently employed in programming to allow data flow across various applications and systems. Many methods and formats, including JSON, XML, Protocol Buffers, and others, can be used to perform serialization. Depending on the needs of the application, such as data size, performance, and compatibility with various programming languages, a serialization format should be chosen. Many advantages of serialization include decreased network traffic, enabled application and system compatibility, and increased scalability of distributed systems. It does have some drawbacks, though, such greater complexity and certain security threats.

## E) Deserilization

The process of transforming a stream of bytes back into an object is called deserialization. Deserialization is frequently used in the setting of artificial intelligence to recover previously serialized models that have been stored to a file or database. Without having to start from scratch, the model may be deserialized and then loaded back into memory to be utilized for prediction or other activities.

Deserialization is a crucial component of artificial intelligence and machine learning. A model's training is frequently a time-consuming and expensive computing procedure. It is feasible to save time and resources by reusing the trained model rather than having to retrain it for each use case by serializing and deserializing models.

Making sure the deserialization procedure is carried out properly is a crucial aspect when working with serialized models. Deserialization mistakes or even security flaws may occur from improper execution of the procedure. In order to guarantee that the serialization and deserialization procedures are effective, it is crucial to properly design and test them.

The work by Jeff Hawkins and Dileep George titled "Hierarchical Temporal Memory incorporating HTM Cortical Learning Algorithms" provides a nice illustration of how serialization and deserialization are used in machine learning. The authors of this paper demonstrate how to store and reload trained models in their HTM framework using serialization and deserialization. The authors can reuse trained models by serializing and deserializing models.

# Results

This Part of the text describes results of your works. There can only be mentioned references, MUST point back to Methods and Intro chapter. No more external references.

Code examples must be provided to demonstrate how to use the algorithm/module. Provide a reference to more unit tests, which show the same in more detail. Also provide all diagrams with comments and reference to unit tests, which generate diagrams.

# Discussion

Conclusion of your work should be precise and concise. How was the project, what is done, what is the result... There can be discussion on further work and direction.

# References

|  |  |
| --- | --- |
| 1 | Y. A. S. H. J. &. C. Y. Cui, "Hierarchical temporal memory for anomaly detection in crowded scenes," *IEEE Transactions on Circuits and Systems for Video Technology,* pp. 385-393, 2016. |
| 2 | D. L. &. D. J. J. Yamins, "Using goal-driven deep learning models to understand sensory cortex," *Nature neuroscience,* 2016. |
| 3 | [Online]. Available: https://www.researchgate.net/publication/282815050\_Enhancement\_of\_Classifiers\_in\_HTM-CLA\_Using\_Similarity\_Evaluation\_Methods. |
| 4 | "Dobric, "Neocortexapi," [Online].," [Online]. Available: https://ddobric.github.io/neocortexapi/. |