 doi:10.20944/preprints201808.0194.v1

Article

An Integrative Systems Model for Oil and Gas Pipeline Data Prediction and Monitoring Using a Machine Intelligence and Sequence Learning Neural Technique

Chinwe Onukwugha [1], and Emmanuel Osegi [2,*]

[1] Federal University of Technology (FUTO), Owerri, Nigeria; onukwugha2000@yahoo.com
[2] National Open University of Nigeria (NOUN); nd.osegi@sure-gp.com
* Correspondence: nd.osegi@sure-gp.com; Tel.: +234-802-583-8364

Abstract: Oil and gas pipeline vandalism is a recurrent problem in oil rich zones of Nigeria and its West African neighbors and remains a challenge for multinationals to set ahead control measures to avert possible damages to operations both in infrastructure and business profit margins. In this paper, an integrative systems model comprising of a machine intelligence technique called Hierarchical Temporal Memory (HTM) and a sequence learning neural network called the Online-Sequential Extreme Learning Machine (OS-ELM) is proposed for monitoring and prediction of pipeline pressure data. The system models the continual prediction of pipeline oil/gas pressure signals useful for secure monitoring and control to avert acts of vandalism in oil and gas installations. The HTM uses a spatial pooler operated in temporal aggregated fashion and is defined as HTM-SP. The OS-ELM technique uses an explicit hierarchical training scheme so that the best cost estimates may be found after a stipulated number of trial runs. We study the performance of three OS-ELM neural activations: the sigmoid (sig), sinusoidal (sin) and radial basis function (rbf) activations. The results indicate improvement factors of 1.297, 1.297 and 1.300 of the HTM-SP over the OS-ELM sigmoid, sinusoidal and radial basis activations respectively.

Keywords: ANN; continual learning; machine intelligence; prediction; vandalism; security; SDR

1. Introduction

One of the most troublesome challenges faced by oil and gas operators in Nigeria particularly the South-South regions have been the recurrent issue of pipeline vandalism. This menace to oil and gas installations often results to the high level of insecurity and disruptions to service found in such regions for which the primary causes have been linked to the militant nature of such areas. The consequences of these disruptions are burst or damaged pipelines, fire and consequent explosions. In this regard, measures such as the use of surveillance cameras, intruder alarms etc, including the use of stern looking militarized personnel have been put in place by industry operators in association with the Government to forestall any threats to oil and gas facilities. Notwithstanding these measures, disruptions to oil and gas operations still remain a possibility. This particular challenge has been attributed to the rigid nature of these security measures which is largely due to the limitation in design and operation. If the pressure flow of oil and gas can be effectively monitored and predicted, then it may be possible by way of automatic control devices to avert the disruptions to service and minimize if not eliminate in its entirety the use of human intervention.

Recent research on real time monitoring has been suggested by several authors such as the use of overhead and underwater surveillance wireless monitoring systems [1-2]. The use of artificial

 doi:10.20944/preprints201808.0194.v1

experts and robotics for smart monitoring and sensing in onshore and offshore oil fields is also a very promising approach as in [3, 4]; a discussion of this line of applications can also be found in [5-7]. More recently, the advantages of cortical-like processing agents have been shown to have the potential for effective continual monitoring of pipeline [8, 9]. These agents use neuroscience and biological ideas closely intertwined with software to implement very useful applications for commerce and industry.

This research paper specifically proposes the use of a variant of an emerging state-of-the-art tool for machine intelligence called the Hierarchical Temporal Memory (HTM) for predictive monitoring of oil and gas pipeline pressure data. The HTM is based on a suite of biological constrained soft-computing techniques called the Cortical Learning Algorithms (CLA) or simply HTM-CLA [10]. HTM-CLA possess the desirable property of online (continual) learning which is vital to systems that change through time and that requires automatic decision making. A model of how the system will perform in a real time system is also presented as an integrative system. Also presented is a comparative study of the performances of the aforementioned technique with a well established technique called the Online Sequential Extreme Learning Machine (OS-ELM). The OS-ELM have been shown to perform relatively well on a number of tasks and is claimed to be extremely fast and superior to some other sequence learning algorithms [11]. However, we show in this study that the proposed HTM technique will on the average outperform the OS-ELM for the task of online (continual) prediction of a real world pipeline pressure data but the differences are not substantial so that the OS-ELM may still have the potential to be used as front-end in the integrated monitoring system.

2. Materials and Methods

2.1. Hierarchical Temporal Memory

Hierarchical Temporal Memory (HTM) is specifically a constrained machine intelligence neural network technique for continual learning tasks [12]; its principle is based on the formation of Sparse Distributed Representations (SDR), the use of the SDR to form invariant representations spatially and temporally, and then learning to make continual predictions from these representations using the theory of biology and neuroscience [12, 13]. Most if not all conventional Artificial Neural Networks (ANN) do not posses all these important functional properties or operations; one important point to note here is that most ANN requires separate training and testing dataset instead of continually learning and predicting on the training dataset.

In a typical HTM network, a Spatial Pooler (SP) stage is used to generate Sparse Distributed Representations (SDR); these SDR is compared with real world sensory input or synthetic sensory-like data to generate a matching set of SDR and then a Temporal Pooler (TP) stage is used for making predictions on the matching SDR set formed by the HTM-SP. These SDR are the basic data structures of any HTM neural network and capture the adaptive learning units used in the neocortex – the seat of intelligence in the brain. The idea of SDR was based on an earlier work on the notion of sparse coding earlier proposed in [14, 15].

An instance of the HTM neuron model is as shown in Figure 1. This neuron model is largely inspired by neuroscience studies describing activity-dependent synaptogenesis – a theory which proposes that the growth and origin of biological synapses is stimulated by an external sensory signal [16]. In the diagram of Fig.4, the proximal synapses is designated as green blobs; these blobs are linearly summed to produce a feed-forward activation while a set of corresponding distal

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 9 August 2018 doi:10.20944/preprints201808.0194.v1

synapses are designated by segments of blue blobs describing feedback and context experiences that are or-ed (logically summed) to generate a spiking neural activation when they exceed a pre-specified recognition threshold (denoted by a Sigma sign). There is the notion that feedback and context experiences are formed using the distal connections.

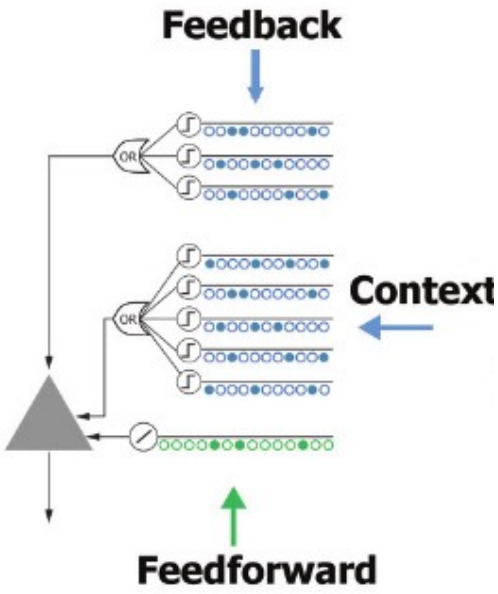

Figure 1. A typical HTM Neuron Model: adapted from [17].

2.1.1. Spatial Pooling in HTM

In HTM, spatial pooling is performed using the notion of SDR followed by competitive Hebbian learning rules, a Homeostatic excitability control, and an overlapping mechanism for deriving candidate or winner SDR patterns via inhibition [17]. SDR are formed by activating or deactivating a set of potential synapses or connecting neuron links. These synapses are grouped into a set of mini-columns and are spread out in a hypercube based on a set of predefined rules.

Let us consider a group of mini-columns with a set of potential connecting logical synapses or neurons. These potential connections may be initialized accordingly as in Equation (1):

$$\Pi_i = \left\{ j \mid \mathrm{I}\left(x_j ; x_i^c, \gamma\right) \ \& \ \mathrm{Z}_{ij} < \rho \right\} \tag{1}$$

where,

j = HTM neuron location index in the mini-column

i = mini-column index

x_j = location of the jth input neuron (synapses) in the input space

x_i^c = location centre of potential neurons (synapses) of ith mini-column in a hypercube of input space

γ = edge length of x_j

 doi:10.20944/preprints201808.0194.v1

ρ = fraction of inputs within the hypercube of input space that are potential connections

Z_{ij} = represents a uniformly distributed random number between 0 and 1

I = an indicator function

The indicator function is usually described by a logical conditioning rule as in Equation (2):

$$\mathrm{I}\left(x_j; x_i^c, \gamma\right) = \begin{cases} 1, & if \;\; x_j \subset x_i^c \\ 0, & otherwise \end{cases} \tag{2}$$

If a set of connected synapses are described in terms of a binary matrix, W, then its formation may be computed by conditioning its associated synapses to a permanence activation rule as:

$$W_{ij} = \begin{cases} 1, & if \;\; D_{ij} \geq \theta_c \\ 0, & otherwise \end{cases} \tag{3}$$

where,

D_{ij} = independent and identically distributed (i.i.d) dendrite synaptic permanence values from the jth input to the ith mini-column

θ_c = synaptic permanence threshold

The i.i.d synapse permanence values are described by Equation (4) as:

$$D_{ij} = \begin{cases} U(0,\ 1), & if \;\; j \in \Pi_i \\ 0, & otherwise \end{cases} \tag{4}$$

Where a natural topology exists, neighborhood mini-columns may be inhibited in accordance to the relation given in Equation (5) otherwise a global inhibition parameter is used.

$$N_i = \left\{ j \mid \left\| y_i - y_j \right\| < \phi,\ j \neq i \right\} \tag{5}$$

where,

y_i = is the ith HTM-SP mini-column

y_j = is the jth HTM-SP mini-column

i, j = mini-column indexes

ϕ = inhibition radius control parameter

For creating associations with input patterns, feed-forward inputs to each of the generative spatial mini-columns are computed using a matching technique called the overlap; this concept is diagrammatically illustrated in Figure 2. The overlap is computed as:

$$o_i = b_i \sum_j W_{ij} z_j \tag{6}$$

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 9 August 2018 doi:10.20944/preprints201808.0194.v1

where,

b_i = is a positive boost factor for exciting each HTM-SP mini-column

z_j = input pattern vector seen by the generative HTM neuron

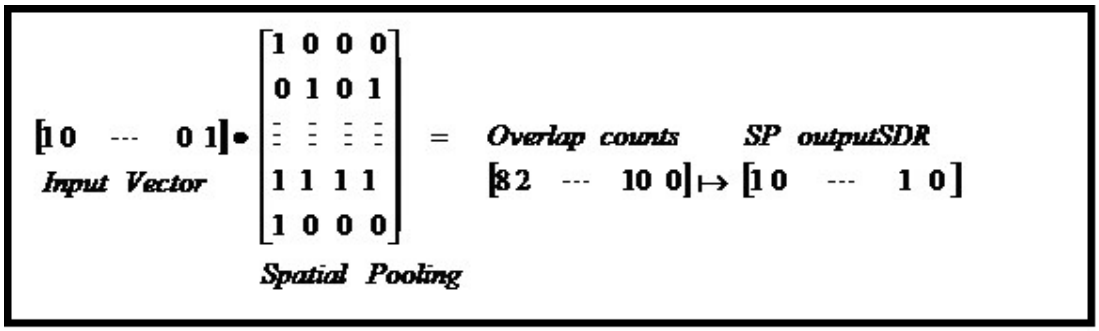

Figure 2. An illustrative concept of the Overlap in an HTM-SP; adapted from [18].

From Equation (6), we can calculate the activation of each SP mini-column as:

$$a_i = \begin{cases} 1, & if\ o_i \geq Z(V_i, 100 - s)\ \&\ o_i \geq \theta_{stim} \\ 0, & otherwise \end{cases} \tag{7}$$

$$V_i = \{o_i \mid j \in N_i\} \tag{8}$$

where,

s = target activation density (sparsity)

Z = a percentile function

θ_{stim} = a stimulus threshold

The HTM-SP uses a learning rule inspired by competitive Hebbian learning for reinforcing dendrite permanence values [17]. The learning rule can be calculated from the formula given in Equation (9) as:

$$\Delta D_{ij} = p^+ D_{ij} \circ A^{t-1} - p^- D_{ij} \circ (1 - A^{t-1}) \tag{9}$$

where,

p^+ = positive permanence value increment

p^- = negative permanence value increment

A^{t-1} = activation state at time step, t

Finally, boost updating in HTM-SP follows the homeostatic excitability control mechanism comparable to that observed in cortical neurons [19]. Boosting is accomplished in HTM-SP using the expressions in Equation (10).

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 9 August 2018 doi:10.20944/preprints201808.0194.v1

$$\bar{a}_i(t) = \frac{(T-1)*\bar{a}_i(t-1)+a_i(t)}{T},\ <\bar{a}_i(t)>=\frac{1}{|N_i|}\sum_{j\in N_i}\bar{a}_j(t),\ b_i = e^{-\beta(\bar{a}_i(t)-<\bar{a}_i(t)>)} \quad (10)$$

where,

$\bar{a}_i$ = time averaged activation over the last T SDR inputs,

T = an integer number denoting the number of Monte Carlo trials to obtain a reasonable activation estimate.

$a_i(t)$= the current activity of the ith mini-column at time step t.

β = a positive parameter that controls the strength of the adaptation effect
As mentioned in [17], "such calculations have been used in previous models of homeostatic synaptic plasticity" as in [20, 21].

2.1.2. Temporal Classifier

In the proposed HTM system, feed-back associations are built from the HTM Spatial Pooler (SP) SDR using a temporal overlap classifier. The Temporal classifier uses the overlap technique which is similar to Equation (6); however predictions are made by performing a match between a set of past observation sequences in the SDR (used as context) and its current observation sequence. The temporal overlaps through time are obtained using Equation (11).

$$o_{j_t} = \sum_{j_t} W^{sp}_{j_t} W^{sp}_{(k-N_c):j_t},\quad N_c < k \le j_t, \quad (11)$$

where,

N_c = Number of past sample sequences in the SDR used as context

k = size of the temporal aggregated (bi-variate) sequence of the SDR through time

j_t = temporal aggregation index number

$W^{sp}_{j_t}$ = the bi-variate SDR after temporal aggregation

2.1.3. Temporal Classifier

A Temporal Aggregation Technique (TAT) is used in the HTM-SP to build a cause-and-effect data sequence from a univariate integer sequence of the SDR formed in the spatial pooling step and then used for an overlapping temporal classification (OTC); the assumption here is that the previous sequence is the cause of the next sequence. The proposed HTM-SP technique is a variant of a software tool developed earlier in [22] and can be found in [23]. In HTM-SP, adding more variables increases the degree-of-freedom for making effective overlap matches. The temporal aggregation procedure used in the predictions is as follows:

- Form a single-column vector matrix of length 1:N having a with a width of 1, where N represents the number of sampled sequences of the SDR obtained from the HTM-SP stage. The elements in this matrix contain the indexes for temporal aggregation.

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 9 August 2018 doi:10.20944/preprints201808.0194.v1

- For each element in the matrix formed in Step 1 greater than 1, perform a modulus operation such that if a remainder exists for the considered element we skip that element, otherwise we select the element; this operation results in single-column vector matrix of length approximately equal to 1:$N/2$. The elements in this matrix contain the set of even indexes in the matrix obtained from Step1 at time instance, t. We call this set $A_{t(1)}$.

- For all elements in the set $A_{t(1)}$, form a concatenation of $A_{t(1)}$ with $A_{t(1)}$ 1-step behind as $\{A_{t(1)} A_{t-1(1)}\}$; this concatenation represent the temporal aggregator index set. We call this set of indexes $A_{t(agg)}$.

- Using $A_{t(agg)}$ as index sequence, extract SDR patterns obtained from the HTM-SP stage in a temporal aggregated fashion and then perform overlap temporal classification through time.

2.2. The Online Sequential Extreme Learning Machine

In standard neural network paradigm, learning data sequence(s) or block(s) online implies continually learning the input-output associations sequence by sequence or chunk-by-chunk, in such a way and manner that the predictions have to vary at each time step and are formed as anew at next time step. The Online Sequential Extreme Learning Machine (OS-ELM) is one neural machine learning technique that conforms to this concept. The OS-ELM technique was developed in [11] and has been shown to be faster with good generalization over a similar sequence learning network called the Resource Allocation Network (RAN) proposed in [24] and its associated variants [23-29]. However, from experience with working with the OS-ELM on real world data, we discovered that OS-ELM may not perform favorably well generating models with large standard errors particularly for univariate series i.e. sequential data without any associated labels. These large deviations may be the result of local learning. In the following sub-section (sub-section 2.2.1) we propose an explicit hierarchical training scheme that affords a global solution space.

2.2.1. Explicit Hierarchical Training of the OS-ELM Neural System

As mentioned previously, the OS-ELM may suffer from a local learning bias. To perform a search for a global solution space and generate reliable predictions, an explicit hierarchical training scheme is used based on a revised OS-ELM functional class coined OSELM_{rev} which includes a Mean Absolute Percentage Error (MAPE) update to calculate the test prediction errors. The original OS-ELM functional class can be found in [30]. However, this scheme basically uses a first order simulation run to generate predictions which encourages local learning but this may likely hamper knowledge discovery.

In order to overcome the aforementioned limitation, an n-order simulation is proposed here which is based on the aforementioned scheme. This encourages the discovery of a global prediction space and requires that several meta-iteration runs akin to a Monte Carlo simulation be made to obtain a best fitted prediction sequence set. By best fitted prediction is meant that single column-wise prediction sequence from a multitude of columnar prediction sequences that gave the least MAPE value.

We perform the following algorithmic steps to derive our global prediction space as follows:

Algorithm 1.

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 9 August 2018 doi:10.20944/preprints201808.0194.v1

Start:
Initialize all OS-ELM System Variables
Define a Maximum Global Iteration Count, C_{max}
    For each iteration in C_{max}, compute the function, $OSELM_{rev}$
    Extract the errors $MAPE_{error}$
    Extract the Test Prediction Sequences as a columnar matrix set, Y_{pred}
Compute the Mean Error as μo of $MAPE_{error}$
Compute the Minimum Error as $MAPE_{min}$ of $MAPE_{error}$
    For all errors in $MAPE_{error}$, find the index ($best_{index}$) that gives least error, $MAPE_{min}$
    Extract the Best Test Prediction, $Y_{pred(best)}$ from Ypred constrained by $MAPE_{min}$
End

2.3. Modelling the Pipeline Monitoring System

2.3.1. Pipeline dataset and Label Encoding

The OS-ELM is very sensitive to the way the data is presented to it. If data is not well labelled, it may generate results that are very far away from the expectation even with small values of errors. Thus, care must be taken to ensure that a univariate sequence does not hamper the OS-ELM performance. For the HTM-SP this is not a problem as the Temporal Aggregation Technique (TAT) described in a previous sub-section (sub-section 2.1.3) takes care of this limitation. A sample scheme of the encoding process for the OS-ELM is as shown in Table 1. In this table, the first column represents the pressure (sequence) signals obtained from a pipeline pressure sensor while the second column is the helper label code – in this case the time stamp in hours. The sequence signals may also be any other measurable physical quantity as long as it can be sequentially monitored. The entire dataset used for the simulations is provided in Appendix A.

Table 1. A sample encoding scheme for the pipeline dataset

Sequence Signal Value	Hourly Time Stamp
17	06.00
16	08.00
18	10.00

2.3.2. Integrative Systems Model

A schematic model of how the proposed techniques may be incorporated into a system for real time monitoring is as shown in Figure 3. The system incorporates an input DATA SOURCE block that functions as pipeline pressure signal sensing unit, a COMBINER block that concatenates time labels with pressure signals, an OS-ELM PROCESSOR block, an HTM-SP PROCESSOR block, two operator (switching) blocks and a CONTROL/ALERT block.
The first operator block (OP-1) is used for selecting the appropriate processor (OS-ELM or HTM-SP) block that feeds to the CONTROL/ALERT block through a second operator block (OP-2) i.e. the processor block that gives least error will pass the predicted signal that activates the CONTROL/ALERT block via OP-2 block. The CONTROL/ALERT block will typically be activated once the pressure line drops below a threshold pressure value which may be defined explicitly or by a reference pressure signal tapped from a consensus of continually predicted pressure signals as

suggested in [31] for monitoring temperature sequences; these aforementioned operations are performed by the second operator block (OP-2).

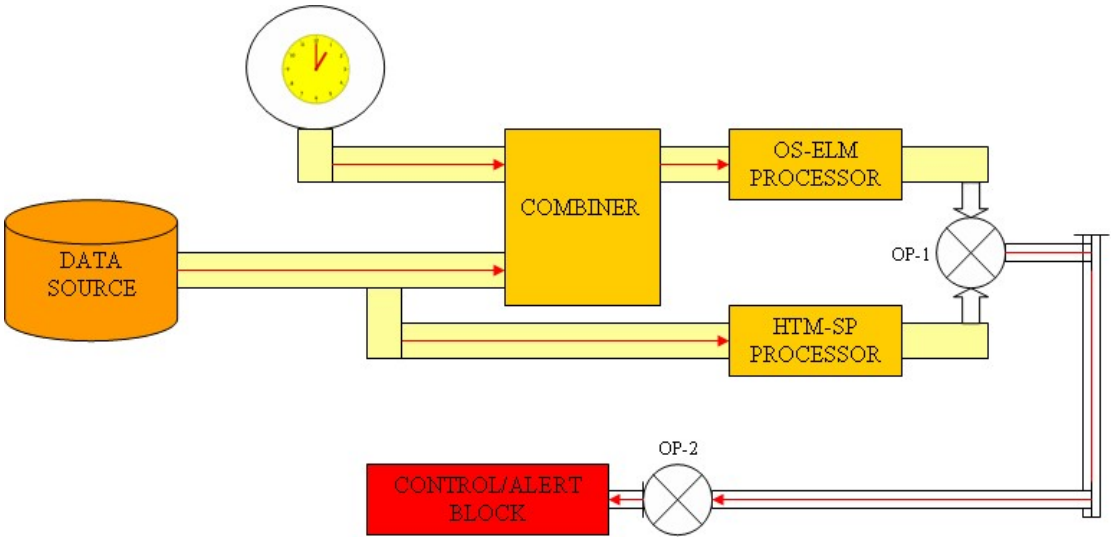

Figure 3. An Integrative Systems Model for real time pipeline monitoring and control

3. Results

The continual learning/predictive performance of the Hierarchical Temporal Memory Spatial Pooler (HTM-SP) is presented and also compared with the Online Sequential Extreme Learning Machine (OS-ELM).

3.1. Performance metrics and parameter tuning

For comparing the results we use the Mean Absolute Percentage Error (MAPE) to evaluate the predictions in both techniques. The MAPE is less sensitive to outliers particularly for large values in the predictions. The MAPE is computed as:

$$MAPE = \left(\frac{\sum_{i}^{n} (| y_{(i)} - \hat{y}_{(i)} |}{n_z} \right) * 100, \qquad i = 1,2,3...,n \quad n \in Z \tag{12}$$

where,

y = the observed input data sequence

$\hat{y}$ = the model's predictions of y

 doi:10.20944/preprints201808.0194.v1

n_z = size of the observation matrix i.e. rows multiplied by columns, and

n = number of the observations in the data or prediction sequence.

We also used a novel evaluation metric called the improvement factor that has been proposed in [32] for comparative evaluations between the estimated fitness values of a proposed and existing algorithm(s) while using the proposed as reference base.

In our case, we have redefined this factor without any loss in generality as:

$$M_{improve} = \frac{M_{existing}}{M_{proposed}} \quad \{M_{proposed} > 0 \tag{13}$$

where,

$M_{existing}$ = mean MAPE value of the existing algorithm or technique

$M_{proposed}$ = mean MAPE value of the proposed algorithm

If $M_{improve}$ is less than or equal to unity, then there is no improvement otherwise there is an improvement of the proposed over the existing technique.

3.1.1. Adjusting the OS-ELM parameters

For the OS-ELM to give appreciable error values its default parameters must be adjusted. This process is called parameter tweaking. We have adjusted three OS-ELM parameters: the percent training, the number of hidden neurons and the activation function parameters. In a tweaking process, one parameter is varied while the other two are kept at default or pre-specified values. This process is repeated until all parameters have been tweaked to meet a reasonable desired performance level. The default key system parameters used in the OS-ELM analysis after some initial tweaking is given in Appendix B (Table B1).

3.1.2. Adjusting the HTM-SP parameters

As mentioned previously, only a single parameter is adjusted in the HTM. This parameter allows the HTM-SP to make continual single or multi-step forecasts. We have considered the HTM-SP look-ahead parameter as this has a profound effect on the HTM-SP performance predictions. All other HTM-SP parameters remain at their default values. The default key system parameters used in the HTM-SP analysis are given in Appendix B (Table B2).

3.2. Performance results using variants of the OS-ELM activations

The best fitted average Mean Absolute Percentage Error (MAPE) performance of the OS-ELM for three activation functions: the sigmoid (sig), sinusoidal (sin) and radial-basis-function (rbf) activations for 5 meta-trial runs at a hidden neuron size of 100 neurons are as shown in Table 2 (averages highlighted in bold). The results indicate that the *sig* and *sin* activations give on the average equivalent MAPE values while the *rbf* activation gave a slightly higher MAPE value. However, in this case, a lower MAPE value does not necessary translate to better performance as the *rbf* generalizes better than the *sig* and *sin* activations.

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 9 August 2018 doi:10.20944/preprints201808.0194.v1

Table 1. MAPE Performance of OS-ELM activation function variants.

Trial No.	$MAPE_{sig}$	$MAPE_{sin}$	$MAPE_{rbf}$
1	0.2983	0.2983	0.2988
2	0.2983	0.2983	0.2987
3	0.2983	0.2983	0.2996
4	0.2983	0.2983	0.2989
5	0.2983	0.2983	0.2988
	0.2983	**0.2983**	**0.2990**

3.3. Continual prediction performance of the HTM-SP

The HTM-SP performance result when the look-ahead parameter is set to 1 and 2 is as shown in the moving MAPE plot of Figures 4 and 5 respectively. In Figure 4, the MAPE values averages at around 0.23 with a peak value of about 0.32 and minimum value of 0.14 while in Figure 5 the corresponding average, peak and minimum values are 0.24, 0.36 and 0.05 respectively. It is possible to achieve much lower values by tweaking the other HTM-SP parameters but we do not consider this form of experimentation.

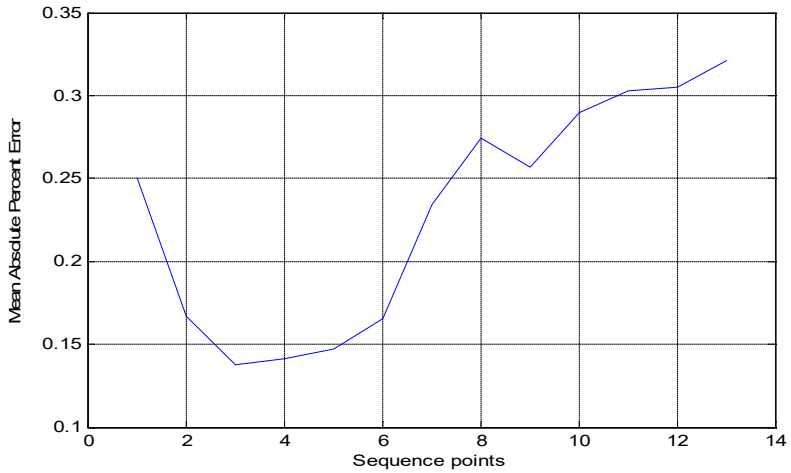

Figure 4. Moving Average plot of the HTM-SP at look-ahead of 1-step

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 9 August 2018 doi:10.20944/preprints201808.0194.v1

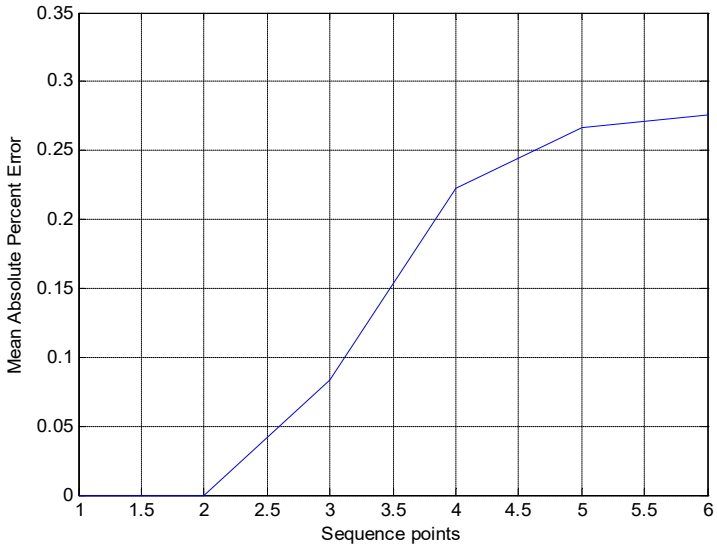

Figure 5. Moving Average plot of the HTM-SP at look-ahead of 2-step

3.4. Comparative Performances of both techniques

The comparative performances of the HTM-SP with the OS-ELM considering all three activations and using the improvement factor metric introduced earlier in subsection 3.1, Equation (13) is as shown in Table 3. In this case, if we propose the HTM-SP as our candidate technique, the results clearly indicates that the HTM-SP will outperform the OS-ELM by a factor of 1.297, 1.297, and 1.300 over the OS-ELM using the *sig*, *sin* and *rbf* activations respectively. Note that reversing the candidacy will obviously not give the OS-ELM any advantage.

Table 1. Improvement of the HTM-SP over the OS-ELM activation function variants.

Improvement Factors		
$MAPE_{sig}$/ $MAPE_{HTM\text{-}SP}$	**$MAPE_{sin}$/ $MAPE_{HTM\text{-}SP}$**	**$MAPE_{rbf}$/ $MAPE_{HTM\text{-}SP}$**
1.297	1.297	1.300

4. Discussion

In this research paper, we have proposed a machine intelligence system called the Hierarchical Temporal Memory (HTM) for real-time monitoring and prediction of pressure signals in a production field useful for securing oil and gas pipelines. We have compared our technique with the Online Sequential Extreme Learning Machine (OS-ELM) and have shown through the MAPE error performances that our technique betters the existing OS-ELM technique. The disadvantage of the OS-ELM is the requirement for training and testing data and in the need for extensive tweaking of several parameters. In the proposed HTM technique, there is no need for such requirement and we have to only tune one parameter. This is an obvious advantage when dealing with real-time automatic learning/monitoring and control system.

The improvement factor metric introduced in this research indicates that the HTM-SP machine intelligence technique is slightly superior over the machine learning one based on the used dataset.

 doi:10.20944/preprints201808.0194.v1

However, this is not to say the OS-ELM did not farewell either. Also the OS-ELM *sig* and *sin* activations are highly stable compared to the *rbf* activation that seem to generalize better.

5. Conclusions

In this research paper, we have proposed an integrative systems approach to the monitoring of real world oil and gas pipeline data using techniques of machine intelligence and machine learning. These techniques include an emerging machine intelligence technique for streaming data analytics called the Hierarchical Temporal Memory (HTM) and a sequence learning neural network technique called the Online Sequential Extreme Learning Machine (OS-ELM).

The HTM technique used a spatial pooling process coined (HTM-SP) with temporal aggregation of learned sparse distributed sequences to make its predictions whereas the OS-ELM technique employed an explicit hierarchical training scheme to compute a global prediction space.

The performances of both techniques as real time security monitoring agents in oil and gas installations are promising. Future research directions should include the integration with Real Time Embedded Operating Systems (RTOS) and a study of the computational cost with the use of full and scaled down versions of both techniques to enhance computational processing speed.

Author Contributions: Conceptualization, E.O.; Methodology, E.O.; Software, E.O.; Validation, C.O and E.O.; Formal Analysis, E.O; Investigation, E.O.; Resources, C.O and E.O; Data Curation, E.O.; Writing-Original Draft Preparation, E.O; Writing-Review & Editing, C.O and E.O; Visualization, C.O and E.O; Supervision, C.O; Project Administration, E.O.

Funding: This research received no external funding

Acknowledgments: The authors will like to thank the field engineers in the Shell Petroleum Development Corporation (SPDC) Flow station Agbada I/II for their kind support and for the provision of real time field data.

Conflicts of Interest: The authors declare no conflict of interest.

Appendix A

Table A1 includes a real world field dataset obtained from a pressure sensor attached to an oil/gas pipeline in the Agbada II flowstation, located at the outskirts of Port-Harcourt city Rivers State Nigeria. This dataset is a two-hourly reading of the pressure sensor. The pressure readings were read and recorded by an onsite Engineer from 06:00a.m to 04:00a.m the next day and for four consecutive days.

Table A1. Two-hourly Pressure Readings of an oil/gas pipeline

Day/Month/Year	Time (Hrs)	Pressure (Bar)
20/02/2017	06:00	17
20/02/2017	08:00	16
20/02/2017	10:00	18
20/02/2017	12:00	20
20/02/2017	14:00	19
20/02/2017	16:00	22
20/02/2017	18:00	19
20/02/2017	20:00	20
20/02/2017	22:00	20

 doi:10.20944/preprints201808.0194.v1

20/02/2017	00:00	20
20/02/2017	02:00	19
20/02/2017	04:00	19
21/02/2017	06:00	20
21/02/2017	08:00	20
21/02/2017	10:00	17
21/02/2017	12:00	21
21/02/2017	14:00	20
21/02/2017	16:00	16
21/02/2017	18:00	15
21/02/2017	20:00	15
21/02/2017	22:00	18
21/02/2017	00:00	18
21/02/2017	02:00	17
21/02/2017	04:00	18
22/02/2017	06:00	17
22/02/2017	08:00	15
22/02/2017	10:00	15
22/02/2017	12:00	16
22/02/2017	14:00	16
22/02/2017	16:00	15
22/02/2017	18:00	11
22/02/2017	20:00	15
22/02/2017	22:00	18
22/02/2017	00:00	18
22/02/2017	02:00	18
22/02/2017	04:00	16
23/02/2017	06:00	15
23/02/2017	08:00	15
23/02/2017	10:00	18
23/02/2017	12:00	12
23/02/2017	14:00	12
23/02/2017	16:00	11
23/02/2017	18:00	12
23/02/2017	20:00	13
23/02/2017	22:00	13
23/02/2017	00:00	13
23/02/2017	02:00	16
23/02/2017	04:00	15

 doi:10.20944/preprints201808.0194.v1

Appendix B

Table B1. OS-ELM Parameters

Parameter Name	Default value
Percent train	50%
Number of Hidden Neurons	100
Activation Function	sig
Number of Iterations	100

Table B2. HTM-SP Parameters

Parameter Name	Default value
Number of Monte Carlo Runs	2
Connected Permanence	0.15
Permanence Increment	0.10
Permanence Decrement	0.01
Number of Samples Used as context	1

References

1. Obodoeze, F. C.; Inyiama, H. C.; Idigo, V. E. Wireless sensor network in Niger Delta oil and gas field monitoring: The security challenges and countermeasures. *International Journal of Distributed and Parallel Systems* **2012**, *3*, 65-77.
2. Obodoeze, F. C.; Nwobodo, L. O.; Nwokoro, S. O. Underwater Real-Time Oil Pipeline Monitoring Using Underwater Wireless Sensor Networks (Uwsns): Case Study Of Niger Delta Region. *sensors* **2015**, *2*, 3597-3602.
3. Shukla, A.; Karki, H. Application of robotics in onshore oil and gas industry—A review Part I. *Robotics and Autonomous Systems* **2016**, *75*, 490–507, doi:10.1016/j.robot.2015.09.012.
4. Shukla, A.; Karki, H. Application of robotics in offshore oil and gas industry— A review Part II. *Robotics and Autonomous Systems* **2016**, *75*, 508–524, doi:10.1016/j.robot.2015.09.013.
5. Castellanos, V.; Albiter, A.; Hernández, P.; Barrera, G. Failure analysis expert system for onshore pipelines. Part – I: Structured database and knowledge acquisition. *Expert Systems with Applications* **2011**, *38*, 11085–11090, doi:10.1016/j.eswa.2011.02.153.
6. Castellanos, V.; Albiter, A.; Hernández, P.; Barrera, G. Failure analysis expert system for onshore pipelines. Part-II: End-User interface and algorithm. *Expert Systems with Applications* **2011**, *38*, 11091–11104, doi:10.1016/j.eswa.2011.02.154.
7. Jawhar, I.; Mohamed, N.; Al-Jaroodi, J.; Zhang, S. An efficient framework for autonomous underwater vehicle extended sensor networks for pipeline monitoring. In *2013 IEEE International Symposium on Robotic and Sensors Environments (ROSE)*; 2013; pp. 124–129.

8. Osegi, E. N. p-DLA: A Predictive System Model for Onshore Oil and Gas Pipeline Dataset Classification and Monitoring-Part 1. *arXiv preprint arXiv:1701.00040* **2016**.
9. ANIREH, V. I. E.; OSEGI, E.N. A PREDICTIVE SYSTEM MODEL FOR OIL AND GAS PIPELINE MONITORING AGAINST POTENTIAL THREAT AND VANDALIZATION. *3*, 15, 303-317.
10. Hawkins, J.; Ahmad, S.; Dubinsky, D. Hierarchical temporal memory including HTM cortical learning algorithms. Techical report, Numenta, Inc, Palto Alto http://www. numenta. com/htmoverview/education/HTM_CorticalLearningAlgorithms.pdf **2010**.
11. Liang, N.-Y.; Huang, G.-B.; Saratchandran, P.; Sundararajan, N. A fast and accurate online sequential learning algorithm for feedforward networks. *IEEE Transactions on neural networks* **2006**, *17*, 1411–1423.
12. Hawkins, J.; Ahmad, S.; Purdy, S.; Lavin, A. Biological and machine intelligence (bami). *Initial online release 0.4* **2016**.
13. Cui, Y.; Ahmad, S.; Hawkins, J. Continuous online sequence learning with an unsupervised neural network model. *Neural computation* **2016**, *28*, 2474–2504.
14. Olshausen, B. A.; Field, D. J. Emergence of simple-cell receptive field properties by learning a sparse code for natural images. *Nature* **1996**, *381*, 607.
15. Olshausen, B. A.; Field, D. J. Sparse coding of sensory inputs. *Current opinion in neurobiology* **2004**, *14*, 481–487.
16. Zito, K.; Svoboda, K. Activity-dependent synaptogenesis in the adult mammalian cortex. *Neuron* **2002**, *35*, 1015–1017.
17. Cui, Y.; Ahmad, S.; Hawkins, J. The HTM spatial pooler—a neocortical algorithm for online sparse distributed coding. *Frontiers in computational neuroscience* **2017**, *11*, 111.
18. Ahmad, S.; Hawkins, J. Properties of sparse distributed representations and their application to hierarchical temporal memory. *arXiv preprint arXiv:1503.07469* **2015**.
19. Davis, G. W. Homeostatic control of neural activity: from phenomenology to molecular design. *Annu. Rev. Neurosci.* **2006**, *29*, 307–323.
20. Clopath, C.; Büsing, L.; Vasilaki, E.; Gerstner, W. Connectivity reflects coding: a model of voltage-based STDP with homeostasis. *Nature neuroscience* **2010**, *13*, 344.
21. Habenschuss, S.; Puhr, H.; Maass, W. Emergence of optimal decoding of population codes through STDP. *Neural computation* **2013**, *25*, 1371–1407.
22. Anireh, V. I.; Osegi, E. N. HTM-MAT: An online prediction software toolbox based on cortical machine learning algorithm. *arXiv preprint arXiv:1708.01659* **2017**.
23. HTM SPATIAL POOLER WITH TEMPORAL AGGREGATION - File Exchange - MATLAB Central Available online: https://www.mathworks.com/matlabcentral/fileexchange/68442 (accessed on Aug 8, 2018).
24. Platt, J. A resource-allocating network for function interpolation. *Neural computation* **1991**, *3*, 213–225.
25. Kadirkamanathan, V.; Niranjan, M. A function estimation approach to sequential learning with neural networks. *Neural computation* **1993**, *5*, 954–975.
26. Yingwei, L.; Sundararajan, N.; Saratchandran, P. A sequential learning scheme for function approximation using minimal radial basis function neural networks. *Neural computation* **1997**, *9*, 461–478.
27. Yingwei, L.; Sundararajan, N.; Saratchandran, P. Performance evaluation of a sequential minimal radial basis function (RBF) neural network learning algorithm. *IEEE Transactions on neural networks* **1998**, *9*, 308–318.
28. Huang, G.-B.; Saratchandran, P.; Sundararajan, N. An efficient sequential learning algorithm for growing and pruning RBF (GAP-RBF) networks. *IEEE Transactions on Systems, Man, and Cybernetics, Part B (Cybernetics)* **2004**, *34*, 2284–2292.
29. Huang, G.-B.; Saratchandran, P.; Sundararajan, N. A generalized growing and pruning RBF (GGAP-RBF) neural network for function approximation. *IEEE Transactions on Neural Networks* **2005**, *16*, 57–67.
30. Extreme Learning Machines Available online: http://www.ntu.edu.sg/home/egbhuang/elm_codes.html (accessed on Aug 7, 2018).
31. Osegi, E. N.; Anireh, V. I.; Onukwugha, C. G. pCWoT-MOBILE: A Collaborative Web Based Platform For Real Time Control In the Smart Space. Proceedings of the 11th International Science, Technology, Arts, Education, Management & the Social Sciences (isTEAMs) Conference, Lagos, Nigeria, 2018, 3, 237-250.
32. Anireh, V.I.; Osegi, E.N. ABC-PLOSS: A SOFTWARE TOOL FOR PATH-LOSS MINIMIZATION IN GSM TELECOM NETWORKS USING ARTIFICIAL BEE COLONY ALGORITHM. *International Journal of Swarm Intelligence* (in press).