Memristor-based LSTM neuromorphic circuits for offshore wind turbine blade fault detection

Harry Burton
Department of Physics and Mathematics
University of Hull
Hull, HU6 7RX, U.K.
harry.burton-2016@hull.ac.uk

Dr Jean-Sebastien Bouillard
Department of Physics and Mathematics
University of Hull
Hull, HU6 7RX, U.K.
J.Bouillard@hull.ac.uk

Dr Neil Kemp
School of Physics and Astronomy
University of Nottingham
Nottingham, NG7 2RD, U.K.
Neil.Kemp@nottingham.ac.uk

***Abstract*—The UK offshore wind industry is rapidly growing to meet CO_2 emission targets. However, the main drawback of the offshore environment is the increased cost of maintenance. Artificial Neural Networks (ANN) show great potential to reduce this cost. Long Short-Term Memory (LSTM) is a form of Recurrent Neural Network (RNN) that shows promising results in solving time series-based problems, making them ideally suited for wind turbine condition monitoring. A dedicated circuit for a LSTM-based ANN that uses memristors will allow for more power efficient and faster computation, whilst being able to overcome the von Neumann bottleneck.**

***Index Terms*—LSTM, Long-Short Term Memory, ANN, Offshore wind, Memristor.**

I. Introduction

Offshore wind farms in the United Kingdom are expected to produce a total of 40 GW by 2030 [1] and have ambitious goals of reaching 90 GW by 2050 [2]. Wind farms are often placed offshore due to the more stable and strong wind conditions. [3]. The offshore environment however has its downfalls, the main of which is the increased cost of maintenance for the turbines. This can cost around 25% of the total cost of the turbine. [4]. One of the ways in which this cost can be reduced is through the use of Condition Monitoring (CM). CM for a wind turbine is the process of observing the components of the turbine to identify any changes in the operation that would be indicative of a fault developing. Robust CM methods can lead to a reduction of costs in maintenance due to a reduction in down time [5] [6].

Recently, the use of Artificial Neural Networks (ANN's) for CM has been gaining significant interest due to their ability to solve highly complex nonlinear problems [7]. The future expansion of ANN's depends on the expansion of network depth, which will require a vast amount of vector-matrix multiplications. Large and deep neural networks have been able to handle complex tasks using extensive amount of data with the advent of vector-matrix multiplication acceleration based on the use of graphics processing using (GPU's) [8]. Despite the advantage that GPU's offer of highly parallel computing that is suitable for ANN's, the high power consumption of a GPU is an obstacle that needs to be improved upon, [9]–[12] which prohibits the use for offshore wind.

The memristor (or memory-resistor) was first proposed in 1971 [13] as a two-terminal device that exhibits non-volatile memory whilst also being able to be scaled down to the nanometer range [14]. The use of a memristor as an in-memory compute unit for making smaller and more power efficient circuits that are dedicated hardware for ANN's [15] has recently been gaining significant interest [16]. Such systems are particularly required in remote sensing applications, where there is a need for high performance computing with low energy usage. In the case of wind turbine blades, there is a substantial cost benefit in not having to source energy from the main turbine, so a low-power computing engine, either battery or solar-powered would be advantageous.

II. Background Information

A. Long Short-Term Memory (LSTM)

The recent rapid increase in volume of data from areas such as the Internet of Things (IoT) has led to the need for a more sophisticated form of near-edge smart memory and information processing. [17] One commonly used and efficient tool that is used for real-time contextualisation of information is the recurrent neural network (RNN). Unlike the classical feed-forward networks, a RNN has a feedback connection between nodes and layers that allow for the input of sequences and to predict the future of said sequence. However, the training of a simple RNN can become a difficult task due to this feedback connection. The algorithms used for the weight updates in RNN's are typically gradient descent based, which for these networks can give rise to an exploding or vanishing gradient during the training process. One way in which this issue has been overcome is through the use of gates to control this feedback loop, which has lead to the development of the Long Short-Term Memory (LSTM) cell.

Figure 1a shows the architecture for a generic LSTM cell c_j [18]. The heart of the memory block is a self-contained linear unit s_c also referred to as the constant error carousel (CEC), which protects the LSTM cell from vanishing and exploding gradient problems that are typically associated with RNNs. The input and output gates here consist of activation functions, along with the corresponding weight matrices. The input gate's with weights net_{in} and output y^{in} is capable of blocking irrelevant data from entering the LSTM cell. Similarly, the output gate with its weights net_{out} and output y^{out} shapes the output of the cell y^c. Finally, the forget gate

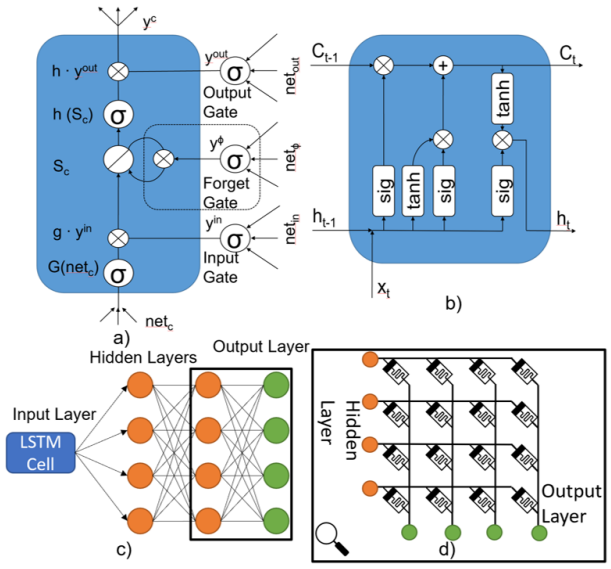

Fig. 1. a) LSTM cell with forget gate, b) Modern representation of an LSTM cell [17], c) Network used for predicting and classifying faults in wind turbines blade, d) Memristor crossbar array used for the neurons in an ANN

highlighted by the dashed line in Figure 1a, which allows the cell to forget or erase information [19]. It can therefore be concluded that the LSTM cell consists of an input layer, an output layer and a self contained hidden layer that is controlled by a forget gate. The mathematical description of an LSTM cell's output for the j^{th} cell at time t can be given by

$$y^{c_j}(t) = y^{out_j}(t)h(s_{c_j}(t)) \tag{1}$$

where $s_{c_j}(t)$ is an internal state of the cell, and y^{φ} is an output for the forget gate, given by

$$s_{c_j}(t) = ss_{c_j}(t-1) + y^{in_j}(t)g(net_{c_j}(t)) \tag{2}$$

$$y^{\varphi_j}(t) = f_{\varphi_j}(net_{c_j}(t)) \tag{3}$$

Since the first proposal of the LSTM cell there have been a variety of different configurations proposed, with other notations for the cell description [20] [21] the most frequent of which is shown in Figure 1b.

B. Issues with Current Computing

In general, ANN have a wide range of applications, with their advancement depending heavily on networks becoming larger with more neurons and layers, and an ever-increasing number of matrix multiplications [22]. The use of graphics processing units (GPU's) has allowed for large and deep neural networks to handle more complex tasks with the use of vast amounts of data [8]. However, the GPU still follows the Von Neumann architecture of classical computers, with separate memory and compute units. This leads to a memory wall problem, or Von Neumann bottle neck, which is mainly due to three aspects [23]: (1) data movement between the processing unit and the memory has a non-negligible latency, (2) data movement in memory hierarchies is limited by bandwidth and (3) a high energy consumption [24]. Furthermore, even with GPU acceleration, a CPU is still needed which can only fetch either data or execute new instruction at any given time, further limiting throughput. This leads to the Von Neumann bottleneck becoming more of a problem as processing speed and memory size increase. One recent way of overcoming this issue is through the use of emerging non-volatile memory (NVM) technologies, more specifically the memristor.

C. Memristor

The memory resistor, or memristor, was first proposed by Leon Chua in 1971 [13] to be the missing link between flux and charge. Recently, the use of memristors for synapse in artificial neural networks has been gaining significant attention [16] [25]. Analogue neural network chips are most commonly made from CMOS technology [15] [26], which have their limitations. A synapse in a neural network must be able to perform three tasks, namely store the weight associated with the synapse, apply this weight to a signal passing through the synapse, and update this weight. A single memristor is able to perform all three of these tasks.

Compared to other implementation of synapses, the memristor has a significant increase in operating speeds. That is, a memristors state can be measured by flowing a current through the device with a low voltage such that the weight of the memristor does not change state. When operating like this, the memristor will appear as a simple resistor, resulting in the device having much higher operating frequencies. It is in this way that the memristor can act as a linear multiplier for a synapse in a neural network [27]. Figure 1d) shows how a memristor crossbar array will be implemented as the synapse for the final network, with the rows correlating to the input from the previous layer and the columns to outputs for this layer.

III. Preliminary Results

To be able to make the final memristor based neuromorphic circuit, the architecture for this specific task must first be determined. An initial computational model will be investigated to determine the optimum structure for the task of classifying and predicting faults in a wind turbines blade. This task can be broken down into two parts, the first being a classifier and the second being able to predict future evolution of faults. In order to investigate the appropriate network for these tasks, a suitable data set is required to train and test such networks. The computational networks in this paper were all made with pytorch, whilst the training of these networks used the Adam optimizer, and a Mean Squared Error loss (MSELoss) function.

A. The Data

The data used to train the networks was from Ou *et al* [28]. In these experiments, a 1.75 m long healthy (i.e. no faults present) wind turbine blade was secured in frame and placed in a humidity controlled environment. The blade was then vibrated and measurements were collected across the blade using a range of different strain gauges and accelerometer as shown in Figure 2. The blade was then subject to a few

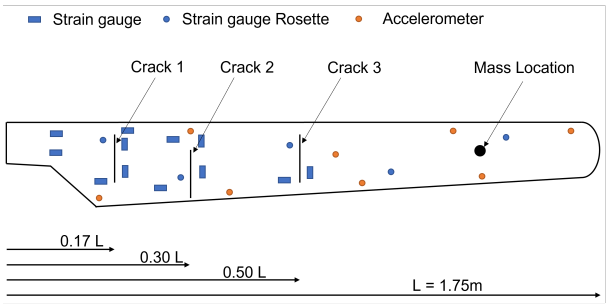

Fig. 2. Sensor Layout on wind turbine blade

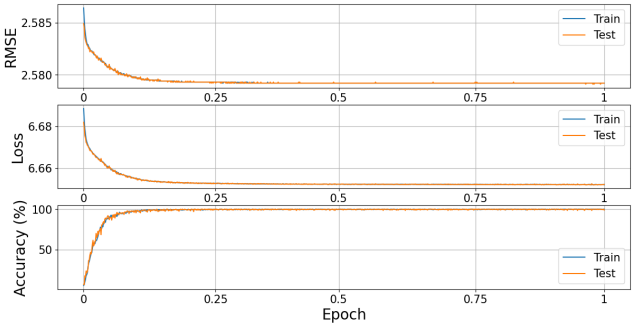

Fig. 3. Classifier's training results, showing that the classifier was accurate after less than one Epoch

different conditions to simulate some different fault cases. First, an increasing amount of mass was added to the blade at the loading position to simulate an increasing amount of icing on the blade. Finally, some cuts were introduced to the blade to simulate cracking. These cracks were all 4 mm deep 1.5 mm wide, with a varying length. The summary of these faults, along with their respective label, can be seen in Table I. This experiment was conducted using one sine wave as the driver for the vibration and four white noise cases for this driver. This data, whilst not exactly operational data, is open source and freely available to use, which allows for a controlled test with known faults present in the blade.

TABLE I
FAULT CASE ON THE BLADE

Case Label	Fault ID	Description		
R	0	Healthy Blade		
A	1	Added mass 1x44g		
B	2	Added mass 2x44g		
C	3	Added mass 3x44g		
D	4	Crack 1: l_1 = 5cm		
E	5	Crack 1: l_1 = 5cm	Crack 2: l_2 = 5cm	
F	6	Crack 1: l_1 = 5cm	Crack 2: l_2 = 5cm	Crack3: l_3 = 5cm
G	7	Crack 1: l_1 = 10cm	Crack 2: l_2 = 5cm	Crack3: l_3 = 5cm
H	8	Crack 1: l_1 = 10cm	Crack 2: l_2 = 10cm	Crack3: l_3 = 5cm
I	9	Crack 1: l_1 = 10cm	Crack 2: l_2 = 10cm	Crack3: l_3 = 10cm
J	10	Crack 1: l_1 = 15cm	Crack 2: l_2 = 10cm	Crack3: l_3 = 10cm
K	11	Crack 1: l_1 = 15cm	Crack 2: l_2 = 15cm	Crack3: l_3 = 10cm

B. The Classifier

To classify faults present on the blade a fully connected network was used. This consisted of an input layer with 26 nodes (equal to the number of inputs from each time step in the data), two hidden layers with 50 nodes each, and an output layer with 12 nodes which corresponds to the number of fault ID's present in the data set. The maximum value for these output nodes was taken to be the classification that the network was predicting, e.g. if the 1^{st} node was the maximum value, this would correspond to the network predicting that the fault ID was 0, if the 2^{nd} node was the maximum value, this would correspond to the network predicting that the fault ID was 1 etc. The nature of these networks means that only a single time step can be used as an input, resulting in the training process being different from the final networks. For this experiment, only the four white noise cases of the data were used. Here, the data was split into time steps such that each point contained the information from all the sensors at an individual time step. This data was then scaled using Equation 4, where x is the input data point, u is the mean of the sample, and s is the standard deviation. This scaling was done to reduce the effect that outliers in the data have on the training process.

$$z = (x - u)/s \tag{4}$$

This was then further split into a test set and a train set, containing 10% and 90% of the overall volume of the data respectively. This data was then used to train the classifier network. Here, batches of the data were fed into the network for a prediction to be made. The error between this prediction and the correct values were then calculated and back-propagated through the network. The results of which can be seen in Figure 3. For this network only a single epoch was needed to reach an accuracy of 100%, whilst it can be confidently concluded that the network did not over fit to the data since the Test line on the plot followed the same trend as the train set and did not deviate. It can therefore be concluded that a simple linear network such as this can be used to accurately classify the faults that are present in the data set used here.

C. The Predictor

The classifier network, whilst good at classifying faults, had no way of taking in a time series, or being able to make classifications/predictions on the future evolution of the blade, and so while being a good tool to implement for detecting faults in the blade, this network would have no use in the predictive maintenance of the blade by itself since no predictions of the future could be made. As such, the LSTM cell was added to the network to allow for a time series to be input to the network and the future of the blade to be predicted. As previously mentioned, the training process for this network differed from that of the simple classifying network since a time series can now be used as an input. To allow for this, some manipulation of the data was needed since in the data, only static faults were present. That is, a fault was introduced to the blade, then the vibrational analysis was done, and then the next experiment was performed. Initially, the icing fault was explored since one experiment will have had only a small impact on the overall health of the blade. Here, a section of

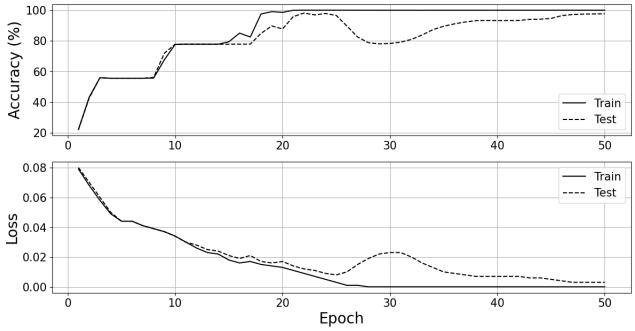

Fig. 4. Training for the LSTM based network, showing the loss and accuracy of the network through its training EPOCHS

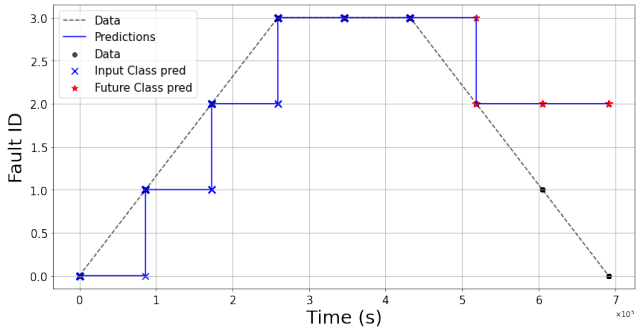

Fig. 5. Prediction of the trained LSTM based network (blue/red) compared to the true fault ID (black)

the healthy blade (case R) was used at the start, then a section of light icing (case A) was added with $86400s$ added to each time step to simulate a day later where some icing had started to form. This was then repeated with the slightly higher icing (case B) and again with the heaviest icing (case C). A further two samples of case C were added, then back down the levels of icing to a blade with no ice in order to create a set of data that consisted of a blade that undergoes a period of ice forming on the blade and then melting, the results of which can be seen by the black line in Figure 5. This was repeated for each of the white noise cases, and then split with one of the white noise cases being for testing the networks and the rest being for training. The data was also scaled using Equation 4.

The structure of the LSTM based network can be seen in Figure 1c. Here, an LSTM cell is used as the input for the network, there are then two hidden layers each with 50 nodes and an output layer with 12 nodes. Similar to the classifier, the output node with the highest value was taken to be the networks prediction for that time step.

The training data was then fed into the network as a whole set, and the network was made to make a prediction. The error was then calculated and back propagated through the network. This was then repeated for a much higher number of epochs than the previous example since here no batches were used. The results of this training can be seen in Figure 4. Here, it can be said that the network had started to learn the patterns in the data after fifty Epochs since the accuracy in both the test and train data sets were high. It can also be fairly confidently said that the network was not over fitting to the data since the validation curve was close to the train curve.

This LSTM based model was then evaluated using the test set of data that was not used for the training process, the results of which can be seen in Figure 5. Here, the true fault ID has been plotted vs time in black to be able to compare the predictions to. The network was given the first two thirds of the data to classify (plotted in blue) and was made to predict the future of the blade (plotted in red), and compare the the true values. This network appears to mostly be able to accurately classify the fault ID at each day, but appears to get the start of each day incorrect and then correct itself to predict the correct value. Similarly, the first day of the future prediction appears to get the initial value incorrect, then correctly predict the fault ID for the rest of the day. The rest of the future predictions however are not correct and the network appears to get stuck predicting a fault ID of 2, i.e. the moderate icing case. This result shows that while the network is able to make short term accurate predictions, more work is needed to discover the correct architecture for this task.

Such an architecture will then be replicated using a neuromorphic approach based on the memristor, more specifically through the use of memristors. This will consist of using memristor crossbar arrays to accelerate the vector matrix multiplications in the LSTM cell [17], and to fully recreate the linear layers of the rest of the network. Here, the biases in the linear layers, or the weight matrix for the LSTM cell, are mapped to the memristor states, the inputs for the LSTM cell and the linear layers correlate to the input voltages applied across the memristor crossbar array columns. For the calculation to take place, the voltage is applied and a current flows through the memrsitors, and get summed through Kirchhoff's law to the resulting output along each column in the crossbar array. The result of the multiplication is therefore a vector of currents through the columns in the array.

IV. Conclusion

The rapidly expanding offshore wind industry in the UK has a dire need for a more cost effective maintenance plan that utilises condition monitoring the enable the optimal scheduling of repairs and general maintenance in order to reduce costs. The use of a LSTM based ANN for this is showing promising results in classifying and predicting the future evolution of faults in the blade of wind turbines. These models however are computationally made on generic computers, meaning large and power inefficient circuits are needed. The development of a dedicated neuromorphic circuit would greatly reduce the size and power consumption of such a circuit, but using commonly used CMOS based components would still encounter the Von Neumann bottleneck. The use of a memristor in such a circuit would allow for in memory computing, overcoming such a bottleneck. This work highlights the promising results in the appropriate architecture for such a circuit, and gives a suitable application for memristors as synapses in an ANN's.

REFERENCES

[1] Roadmap URE. Department of energy and climate change. UK government, First published. 2011;12.

[2] RenewableUK. POWERING THE FUTURE; 2020.

[3] Bergström L, Kautsky L, Malm T, Rosenberg R, Wahlberg M, Capetillo NÅ, et al. Effects of offshore wind farms on marine wildlife—a generalized impact assessment. Environmental Research Letters. 2014;9(3):034012.

[4] Mazidi P, Tohidi Y, Sanz-Bobi MA. Strategic maintenance scheduling of an offshore wind farm in a deregulated power system. energies. 2017;10(3):313.

[5] Helsen J, Devriendt C, Weijtjens W, Guillaume P. Condition monitoring by means of scada analysis. In: Proceedings of European wind energy association international conference Paris; 2015. .

[6] Tchakoua P, Wamkeue R, Ouhrouche M, Slaoui-Hasnaoui F, Tameghe TA, Ekemb G. Wind turbine condition monitoring: State-of-the-art review, new trends, and future challenges. Energies. 2014;7(4):2595-630.

[7] LeCun Y, Bengio Y, Hinton G. Deep learning. nature. 2015;521(7553):436-44.

[8] Chellapilla K, Puri S, Simard P. High performance convolutional neural networks for document processing. In: Tenth international workshop on frontiers in handwriting recognition. Suvisoft; 2006. .

[9] Chi P, Li S, Xu C, Zhang T, Zhao J, Liu Y, et al. Prime: A novel processing-in-memory architecture for neural network computation in reram-based main memory. ACM SIGARCH Computer Architecture News. 2016;44(3):27-39.

[10] Hu M, Strachan JP, Li Z, Grafals EM, Davila N, Graves C, et al. Dot-product engine for neuromorphic computing: Programming 1T1M crossbar to accelerate matrix-vector multiplication. In: 2016 53nd acm/edac/ieee design automation conference (dac). IEEE; 2016. p. 1-6.

[11] Shafiee A, Nag A, Muralimanohar N, Balasubramonian R, Strachan JP, Hu M, et al. ISAAC: A convolutional neural network accelerator with in-situ analog arithmetic in crossbars. ACM SIGARCH Computer Architecture News. 2016;44(3):14-26.

[12] Gokmen T, Vlasov Y. Acceleration of deep neural network training with resistive cross-point devices: Design considerations. Frontiers in neuroscience. 2016;10:333.

[13] Chua L. Memristor-the missing circuit element. IEEE Transactions on circuit theory. 1971;18(5):507-19.

[14] Šuch O, Klimo M, Kemp N, Škvarek O. Passive memristor synaptic circuits with multiple timing dependent plasticity mechanisms. AEU-International Journal of Electronics and Communications. 2018;96:252-9.

[15] Liu J, Brooke M. A fully parallel learning neural network chip for real-time control. In: IJCNN'99. International Joint Conference on Neural Networks. Proceedings (Cat. No. 99CH36339). vol. 4. IEEE; 1999. p. 2323-8.

[16] Snider G. Molecular-junction-nanowire-crossbar-based neural network. Google Patents; 2008. US Patent 7,359,888.

[17] Smagulova K, James AP. A survey on LSTM memristive neural network architectures and applications. The European Physical Journal Special Topics. 2019;228(10):2313-24.

[18] Hocheriter S, Schmidhuber J. Long Short-Term Memory. Neural Comp. 1997;9:1735-80.

[19] Gers FA, Schmidhuber J, Cummins F. Learning to forget: Continual prediction with LSTM. Neural computation. 2000;12(10):2451-71.

[20] Lipton ZC, Berkowitz J, Elkan C. A critical review of recurrent neural networks for sequence learning. arXiv preprint arXiv:150600019. 2015.

[21] Sutskever I, Vinyals O, Le QV. Sequence to sequence learning with neural networks. Advances in neural information processing systems. 2014;27.

[22] Kim H, Kim T, Kim J, Kim JJ. Deep neural network optimized to resistive memory with nonlinear current-voltage characteristics. ACM Journal on Emerging Technologies in Computing Systems (JETC). 2018;14(2):1-17.

[23] Zou X, Xu S, Chen X, Yan L, Han Y. Breaking the von Neumann bottleneck: architecture-level processing-in-memory technology. Science China Information Sciences. 2021;64(6):1-10.

[24] Mittal S. A survey of ReRAM-based architectures for processing-in-memory and neural networks. Machine learning and knowledge extraction. 2018;1(1):75-114.

[25] Kriesel D. A brief introduction on neural networks. 2007.

[26] Marco BG. Microelectronic neural systems: Analog VLSI for perception and cognition. PhD thesis; 1998.

[27] Sharifi MJ, Banadaki YM. General spice models for memristor and application to circuit simulation of memristor-based synapses and memory cells. Journal of Circuits, Systems, and Computers. 2010;19(02):407-24.

[28] Ou Y, Tatsis KE, Dertimanis VK, Spiridonakos MD, Chatzi EN. Vibration-based monitoring of a small-scale wind turbine blade under varying climate conditions. Part I: An experimental benchmark. Structural Control and Health Monitoring. 2021;28(6):e2660.