Time sensitivity and self-organisation in Multi-recurrent Neural Networks

Oluwatamilore Orojo
*School of Science and Technology
Nottingham Trent University
Nottingham NG11 8NS, United Kingdom
oluwatamilore.orojo@ntu.ac.uk

Jonathan Tepper*
Perceptronix Ltd
Avon Way,
Derby DE65 5AE, United Kingdom
jtepper@perceptronix.net

T.M. McGinnity*
Intelligent Systems Research Centre
University of Ulster, Magee Campus,
Derry BT48 7JL, United Kingdom
tm.mcginnity@ulster.ac.uk

Mufti Mahmud
School of Science and Technology
Nottingham Trent University
Nottingham NG11 8NS, United Kingdom
mufti.mahmud@ntu.ac.uk

***Abstract*—Model optimisation is an essential step in modelling. Traditionally this was limited to just parameter tuning. However, today with more insight on the internal dynamics of model architecture, researchers have explored numerous ways to optimise and enhance performance through model extension and development. In this paper, we extend the architecture of the Multi-recurrent Neural Network (MRN) to incorporate self-learning recurrent link ratios and periodically attentive hidden units. We contrast and show the superiority of this extension to the standard MRN for a complex financial prediction task. This superiority is attributed to i) the ability of the self-learning recurrent link ratios to dynamically utilise data to identify optimal parameters of its memory mechanism and ii) the periodically attentive units enabling the hidden layer to capture temporal features that are sensitive to different periods of time. Finally, we evaluate our extended MRNs (Self-Learning MRN (SL-MRN) and Periodically Attentive MRN (PA-MRN)), against two current state-of the-art models (The Long-Short Term Memory and Support Vector Machines) for an eye state detection task. Our preliminary results demonstrate that the PA-MRN and SL-MRN outperform both state-of-the-art models. These results demonstrate that the MRN extensions are suitable models for machine learning applications and this finding would be further explored.**

***Index Terms*—neural network, attentive nodes, self-learning, multi-recurrent neural network**

I. Introduction

Artificial neural networks (ANNs) are powerful machine learning techniques widely applied for problem solving. Specifically, they have been shown to be an effective learning paradigm which provide state-of-the-art performance in a variety of tasks such as biological data mining [1], image analysis [2], anomaly detection [3], disease detection [4], financial forecasting [5], natural language processing [6] and strategic game playing [7].

ANNs are much simplified yet powerful computational models of the neural apparatus underpinning the human brain [8]. It has been shown that ANNs enable effective i) approximation of complex non-linear mappings and handling of 'noisy' signals, ii) prediction without a priori distribution assumptions (which can obscure learning) and iii) adapting and incorporation of new data [9].

Traditionally, ANNs are trained on a given dataset and optimized solely through hyper-parameter tuning for improved performance. For example, Ulbricht's Multi Recurrent Network (MRN) requires the designer to determine both the learning hyper-parameters (such as; learning rate and momentum) and architecture hyper-parameters (such as number of hidden units, number of memory banks for each layer and the weightings for both self- and layer-level links) [10].

Over the last few decades, researchers have sought to further enhance and exploit ANNs. In particular, researchers have identified that endowing ANNs either directly by adjusting their architecture (for example incorporating self-learning, self-organising, internal decays, attentive nodes) or indirectly; in combination with other techniques (such as wavelet, fuzzy learning) have largely enhanced performance.

In this paper, we seek to extend Ulbricht's original MRN to specifically overcome the two key issues of: i) the requirement of designer input to determine layer- and self-recurrent link ratios and ii) gradient descent learning problem and catastrophic interference of this hidden state due to all temporal features being super-imposed onto a single homogeneous hidden feature layer. To resolve the first issue, we will extend the architecture of an adapted version of Ulbricht's proposed model, the MRN by incorporating self-learning link ratios within its memory banks and to identify whether this extension will enhance performance and Finally, to alleviate the issue of temporal pattern interference, we introduce periodically attentive nodes within the hidden layer of Ubricht's MRN. The main contribution of this paper is algorithmic development: we are among the first to exploit algorithm development in the MRN.

The structure of this paper is as follows. A brief summary

The authors would like to thank Nottingham Trent University for their funding (Vice-Chancellor's Doctoral Scholarship).

of self-learning and attentive nodes within neural network is given in Section 2, The methodology is given in Section 3. The proposed model extensions are applied for financial and detection tasks and results are given in Section 4 and Section 5 concludes.

II. BACKGROUND

A. The Multi-recurrent Neural Network (MRN)

The Multi-recurrent Neural Network (MRN) developed by Ulbricht is a powerful dynamic modelling tool with a unique memory mechanism which enables enhanced information processing and signal extraction. Ulbricht [10] and more recently Tepper [11] and Orojo [5] show that the MRN, a slightly more sophisticated class of Simple Recurrent Network (SRN), is better able to capture the latent signal in time-series data and found that the MRN dynamics improved learning and achieved better accuracy (compared to the SRN, NARX, ESN and LSTM networks). However, akin to the numerous ANN variants, the MRN requires parameter tuning by the user which can be a lengthy process. In addition, the MRN is not immune to the gradient descent problem and catastrophic interference popularly prone to Neural Networks. In particular, Ulbricht [10] attempts to deal with the gradient descent problem by employing multiple varying strengths of memory banks to preserve information and prevent the network from 'forgetting'. This has proven to enhance learning and performance as shown by [10]–[12]. However, the MRN configuration does not specifically deal with the simultaneous response of both long and short term components as described in [13] and as a result is still susceptible to the gradient descent problem.

Therefore, we seek to investigate two techniques to mitigate the weaknesses of the MRN namely self-learning and periodic attentiveness.

B. Self-Learning in ANNs

A number of authors such as [14]–[18] have investigated and employed various self-learning schemes in neural networks. For example, Hartstein [15] prove that multiplicative synapses are not mandatory in neural networks and that self-learning occurs as the networks utilize its inputs relative to learnt threshold voltages. Particularly, the learning of the voltage thresholds occurs using a Hebbian rule and these learnt thresholds are believed to contain the memories of these networks. They view and demonstrate the concept of self-learning in their network as the dynamic learning capability of a network irrespective of its synapse function. Conclusively, their work shows that neural networks in and of themselves are robust and possess innate self-learning abilities.

Bouchachia [14], on the other hand, implements the idea of self learning in a Recursive Neural Network by adopting semi-supervised learning for their dataset which is comprised of both labelled and unlabelled data, thus coercing the network to harness its structure to extract and learn from the limited labeled data. Other authors such as [19] and [20] use similar techniques particularly grouping the unlabeled data using a relevant scoring system. Lee [16] further build on this idea of self-learning by combining the following schemes; pre-training, dropout and error forgetting which they show lead to improved performance. In addition, Nguyen [17], show that neural networks can learn independently and demonstrates that limited user input and design is achievable.

Research on self-learning/training in neural networks appears to mainly occur in the 'forward' layers. There is however limited work on self-learning for the context/recurrent layer(s) in recurrent neural networks. These layers are essential for processing temporal data, particularly they serve as the networks 'memory' allowing the network to 'remember' past information which is essential for prediction. In an attempt to bridge this gap, we seek to identify if we can harness the self-learning ability of the neural network within its recurrent layer to identify how and if this will enhance learning and improve performance. We would explore and introduce self-learning in the MRN by endowing the memory within the MRN to understand whether the MRN has the potential to learn specific ratios for the self- and layer-recurrency links for a given task to reduce parameter tuning in the memory layer.

C. Temporal Attentiveness in ANNs

RNNs are notourisly known to suffer from the gradient descent problem. Either they explode where the gradients influenced by the long-term components rapidly increase during training or they vanish rapidly to zero. These behaviours are largely caused by the long-term components in the network [21]. Particularly, all components within the network (that is both short-term and long-term) respond simultaneously to every time period and as such these networks are unable to preserve components for the long-term dependency learning once the short-term dependency learning has occurred [13]. This behaviour is reflective of the synchronous neural firing in the brain for representing information [22]. Additionally, ANNs suffer from catastrophic interference, a significant problem caused by continual learning where the network forgets knowledge acquired from previous tasks (or data distributions) due to connectionist nature of these networks [23], [24].

O'Connell [13] sought to deal with this problem within Simple Recurrent Networks (SRNs) by employing Periodically Attentive (PA) units to extend the temporal capacity of the SRN. Particularly, he found that for learning a long term dependency task such as the embedded reber grammer, the vanilla SRN was unable to learn both the short and long term predictors for the task simultaneously. In addition, given the recurrence of the short-term error, the resources of the network were predominately utilised for the short-term task [13].

These PA units are configured to receive inputs at regular intervals such that they enforces the network to partition its units into long and short term [13]. As a result, even if the network initially learns the short term prediction task (that is most units in the hidden layer are allocated for the short-term task), some units will still be available to learn the long-term prediction task as they will not have been satisfactorily allocated as short term predictors. Thus PA units ensure

appropriate resource allocation for long term dependency tasks [13].

Other researchers such as [25]–[31] have achieved varying success by embedding attentiveness in neural networks. Chaudhari, [32], provides an overview of the varying implementations of attention, neural network models employing attention and in particular discussing the significant gains from attention incorporation in neural networks.

Given the success with embedded attention in neural networks, in this paper, we seek to explore attentiveness in a neural network variant to encourage long-term dependency learning and fully exploit the ability of the MRN. We will extend the methodology in [13] to the MRN in an attempt to i) overcome the vanishing gradient problem by leaving node information from previous time periods active for longer (which is then further embedded within the memory banks) and ii) reduce catastrophic interference by dedicating hidden units to specific phases of time.

III. Methodology

A. Data and Forecasting Methodology

The proposed models will be benchmarked on two datasets in different domains to assess their performance and transferability (that is their ability to generalise in different context/domains).

1) Oil price data: The dataset and forecasting methodology employed is detailed in [5]. In this paper, the best window size (300) presented in [5] is employed.

2) Electroencephalogram (EEG) data: EEG data created by Rösler and Suenderman [33] for eye state detection was used as a benchmark. There are 14,980 instances of eye state with each instance consisting of 14 EEG features and an eye state class/label (0: open (55%), 1: closed (45%)). The EEG features were standarised using the mean and standard deviation.

B. Models

Multi-recurrent Neural Networks (MRNs) originally proposed by [10] utilise a combination of repeated memory banks (these are feedback activations from one or more layers in the network's architecture) with varying strengths. Particularly, the composition in the memory banks is determined by a ratio of layer-level recurrency and self-recurrency of the number of memory banks.

1) Introducing Self Learning in the MRN: The MRN is configured with additional hidden units which act as the Ratio Control Units (RCUs) that determine the layer-and self-recurrency ratio links in the memory banks. Thus the hidden units in the proposed MRN are larger than in a vanilla MRN. These additional units are treated like the other hidden units but are specifically randomly initialised in the half-open interval [0.0, 1.0). After a forward pass and/or backpropagation (when the errors are sent back through the network in order to minimize the cost function by adjusting the network's parameters (weights, and biases)), the RCUs are updated and used for copying the new available information. A modified MRN architecture is illustrated in Fig. 1 where the RCUs are represented as the black hidden units.

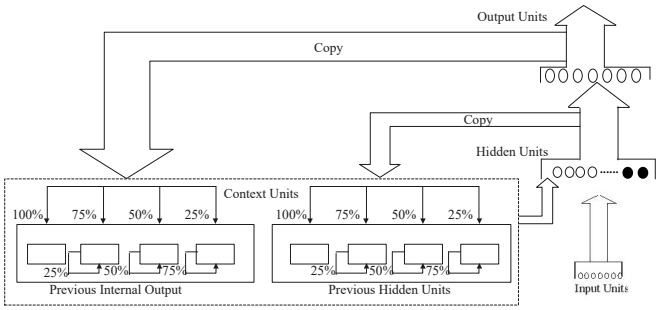

Fig. 1. MRN with self-learning links (adapted from [5])

The modified MRN functions are as follows. Given the inputs at time t, I_t, the hidden units at time $t-1$, H_{t-1} and the hidden memory at time $t-1$, M_{t-1_h}, the volume of information stored in the hidden memory at time t is calculated as:

$$M_{t_h} = (RCU_{t_h} \times H_{t-1}) + ((1 - RCU_{t_h}) \times M_{t-1_h}) \quad (1)$$

where RCU_{t_h} are randomly initialized and thereafter updated from the allocated hidden units, the output units at time $t-1$, O_{t-1}, the output memory at time $t-1$, M_{t-1_o}, the volume of information stored in the output memory at time t is calculated as:

$$M_{t_o} = (RCU_{t_o} \times O_{t-1}) + ((1 - RCU_{t_o}) \times M_{t-1_o}) \quad (2)$$

where RCU_{t_o} are randomly initialized and thereafter updated from the allocated hidden units and their respective weights to the hidden layer W_{i_h}, $W_{M_{hh}}$ and $W_{M_{oh}}$, the net hidden units at time t are calculated as:

$$\hat{H}_t = \sum W_{i_h} I_t + \sum W_{M_{hh}} M_{t_h} + \sum W_{M_{oh}} M_{t_o} \quad (3)$$

The hidden units at time t are derived by passing the net hidden units at time t through the chosen activation function for the hidden layer (f_h) and the RCUs are updated as:

$$\begin{aligned} H_t &= f_h(\hat{H}_t) \\ RCU_{t_h} &= H_t[u_h : n_h] \\ RCU_{t_o} &= H_t[(n_h + 1) : n_o] \end{aligned} \quad (4)$$

where u_h is the total number of hidden units, n_h is the total number of memory banks for the hidden layer and n_o is the total number of memory banks for the output layer. Given the hidden units H_t at time t and the hidden to output weights W_{ho}, the net output units at time t are calculated and passed through the chosen activation function for the output layer:

$$\begin{aligned} \hat{O}_t &= \sum W_{h_o} H_t \\ O_t &= f_o(\hat{O}_h) \end{aligned} \quad (5)$$

2) Introducing Periodically Attentive (PA) Units in the MRN: The MRN is configured such that PA units receive patterns in a window sequence periodically and non - PA units receive patterns at every time-step (note that with every new window the hidden activations are reset to the mid-point of the hidden unit activation function). Consider a sequence of 12 observations with a time attentive sensitivity of 3 thus we divide the PA units into 3 Time Attentive Periods (TAP) and the remaining units (if any) are responsive to all time periods such that they have an Holistic Time Attentive Period (H-TAP) (that is units that are active at every period). The PA-MRN processes the sequence as shown in Table I *(Note: p_t refers to a given pattern at time t, a phase consists of all TAPs, a minus sign indicates the beginning of a new sequence where the units are set to a known initial value based on the mid-point of the hidden unit activation function and a hat symbol (^) indicates that the hidden unit activation values of the nodes associated with that time period remain fixed/unchanged).*

TABLE I
EXAMPLE OF PERIODICALLY ATTENTIVE UNITS (FOR A WINDOW OF SIZE 12)

Time step	TAP 1	TAP 2	TAP 3	H-TAP	Phase
1	p_1	-	-	p_1	1
2	^	p_2	-	p_2	1
3	^	^	p_3	p_3	1
4	p_4	^	^	p_4	2
5	^	p_5	^	p_5	2
6	^	^	p_6	p_6	2
7	p_7	^	^	p_7	3
8	^	p_8	^	p_8	3
9	^	^	p_9	p_9	3
10	p_{10}	^	^	p_{10}	4
11	^	p_{11}	^	p_{11}	4
12	^	^	p_{12}	p_{12}	4

3) Baseline Models: Two classifiers, Long-Short Term Memory (LSTM) and Support Vector Machine (SVM) are compared to the models presented in this paper.

- **LSTM**: Two different LSTM models with different optimizers (Adam and Root Mean Square Propagation (RMSprop)) are employed. All the LSTM models have 2 LSTM layers and 1 dense layer to enable effective representation. The first LSTM employs a hyperbolic tangent activation function while the second LSTM employs a sigmoidal activation function.
- **SVM**: SVMs are widely used for classification tasks and in this paper we employ three SVM models with different kernels (polynominal, sigmoid and Radial Basis Function (RBF)).

IV. RESULTS

In this section, three variants of the MRN are benchmarked on oil price dataset and comparative results presented. In order to access the transferability and generalisation of these extensions, the models will benchmarked in a different domain for more complex time-series processing using EEG data and compared with current state-of-the-art models; Long-Short Term Memory (LSTM) and Support Vector Machine (SVM). Various combinations of parameters were exploited and the best models were selected.

A. Financial Application

Oil price prediction task is a notoriously challenging yet important task, particularly oil prices play a key role in guiding an economy's trajectory and are a key determinant of a nation's well-being [5]. In this paper, three variants of the MRN are applied for the oil price prediction task and the results are compared. *The MRN models used a sigmoid activation function for the hidden layer and the linear activation function for the output layer.*

1) Vanilla MRN and SL-MRN: The vanilla MRN is compared to the SL-MRN and in Table II the Root Mean Squared Errors are presented. The MRN with self-learning links in general outperforms the vanilla MRN. In particular, it requires a smaller network thus showing the ability of the network to robustly and adequately use the available data to determine optimal ratio links.

Interestingly, it can be seen that using the optimal number of memory banks for the vanilla MRN with the MRN with self-learning links causes a drop in performance. This behaviour is probably due to overfitting where the network begins to learn and fit to the peculiarities of the data instead of learning for the given task. The learned ratio links for the best SL-MRN at different horizons are shown in Table III.

TABLE II
COMPARATIVE NETWORK PERFORMANCE OF THE VANILLA MRN AND MRN WITH SELF-LEARNING LINKS

Horizon	Memory Bank	Vanilla MRN	MRN with self-learning links
1	[0, 5, 4]	0.3679	0.4359
	[0, 3, 0]	0.4109	0.3643
3	[0, 3, 2]	0.7455	0.9789
	[0, 3, 0]	0.8323	0.6970
6	[0, 4, 4]	0.9585	0.9797
	[0, 3, 3]	1.1123	0.9329
12	[0, 4, 3]	1.0152	1.2151
	[0, 4, 2]	1.2894	1.0979

The SL-MRN is particularly suited for complex time series problems as it enables the adjustment of the RCUs based on the error gradients which are calculated from the complex input-output mappings and the networks predictions and this behaviour (that is simultaneous learning) is shown to enhance the network's performance.

2) Vanilla MRN and PA-MRN: The Vanilla MRN is compared to the PA-MRN and the results are presented in Table IV. The table shows the number of PA units used, for example 17 (3) refers to 17 periodically attentive units divided into 3 batches with each batch receiving an input every 3^{rd} time-step and 3 (20 - 17) units attentive to all time-steps. *(Note: all networks had 20 hidden units but memory banks*

TABLE III
LINK RATIOS CONFIGURATION FOR THE VANILLA AND MRN WITH SELF-LEARNING LINKS (*MB stands for Memory Bank*)

	Ratio links for memory banks				
Layer	Input	Hidden		Output	
Horizon	Links	Standard links	Learnt links	Standard links	Learnt links
1 (MB=[0,3,0])	0	0.3333, 0.6667 1	0.8275, 0.6198, 0.5951	0	0
3 (MB=[0,3,0])	0	0.3333, 0.6667, 1	0.1851, 0.1152, 0.2488	0	0
6 (MB=[0,3,3])	0	0.3333, 0.6667, 1	0.5939, 0.6241, 0.1649	0.3333, 0.6667, 1	0.2008, 0.4696, 0.4156
12 (MB=[0,4,3])	0	0.25, 0.5, 0.75, 1	0.0861, 0.4118, 0.0454, 0.1473	0.5, 0.5	0.7988, 0.7632

and window sizes varied to encourage effective modeling).

As seen from Table IV, the PA-MRN outperforms the vanilla MRN for an horizon of 3 and 6 however, for an horizon of 12 fairs comparatively to the vanilla MRN. For an horizon of 1, the vanilla MRN performs better. In particular, during training the PA-MRN had large rapid oscillations between errors and this unstable behaviour likely hindered the prediction performance. Secondly, the PA-MRN needs more hidden units to compensate for the partition of units as the units in some partitions were not sufficient to learn the task. Importantly, for the horizon of 3 and 6, we see the potential of the PA-MRN and with adequate optimisation, the PA-MRN possesses the ability to consistently outperform the vanilla MRN which has been optimised for the task. Finally, the prediction accuracy for the PA-MRN used smaller window sizes (10, 15 and 150) thus showing its robustness to learn effectively from the data. *(Note: large window sizes such as those used with the vanilla MRN (300) led to rapid gradient descent and as a result hindered the learning in the network).* The PA-MRN's effective use of smaller window sizes enables early prediction and indication and this is particularly suited to real-time tasks where limited data is available at any point in time.

TABLE IV
COMPARATIVE NETWORK PERFORMANCE OF THE VANILLA MRN AND PA-MRN

Horizon	1		3	
	MRN	PA-MRN	MRN	PA-MRN
PA Units	0	20 (2)	0	17 (3)
RMSE test set	0.3679	0.3986	0.7455	0.6796
Horizon	6		12	
	MRN	PA-MRN	MRN	PA-MRN
PA Units	0	20 (5)	0	20(5)
RMSE test set	0.9585	0.9023	1.0152	1.09

B. Eye Detection Task

In recent years, there has been an increasing interest in eye state detection particularly for applications such as facial expression recognition, physiological state detection, human-computer interaction, driver fatigue detection and many more [34], [35]. In this paper, a number of classifiers are applied for the eye state detection task that is to determine from EEG signals whether the eyes are open or closed (see Fig. 2 and preliminary results are presented. *The MRN models used the hyperbolic tangent activation function for the hidden layer and the sigmoid activation function for the output layer.*

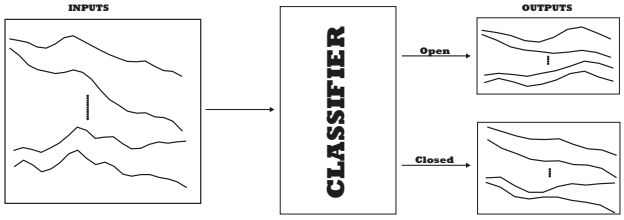

Fig. 2. Eye detection task for the classifiers

In order to assess the performance of the chosen classifiers, the Area Under Curve (AUC) was calculated and presented in Table V, the MRN extensions achieve the best scores. This is due to the extensions i) better utilization of the MRNs architecture to process the data and learn the task which enables superior performance and ii) the PA - MRN particularly ensures the preservation of information through effective unit allocation within the hidden layer.

TABLE V
COMPARATIVE RESULTS FOR ALL THE CLASSIFIERS BENCHMARKED ON THE EEG EYE STATE DATA

Models	AUC score
PA - MRN	0.9
MRN - Self-learning links	0.877
Vanilla MRN	0.854
SVM (sigmoid)	0.85
LSTM (RMSProp)	0.849
LSTM (Adam)	0.826
SVM (polynomial)	0.622
SVM (RBF)	0.555

V. CONCLUSION

The multi-recurrent network has been presented by a number of researchers as a powerful paradigm for a number of machine learning applications however it is prone to limitations akin to similar paradigms. In particular, we address two main limitations of the MRN: i)ad-hoc user design and ii) diminishing gradient and atastrophic interference. The MRN is extended with i) self-learning links to encourage dynamic learning and reduced user design and ii) periodically attentive nodes to ensure the network learns all tasks (short and long) and preserves information. The MRN extensions are

compared to the MRN for a financial task and both extensions demonstrate superiority (in performance and applicability) to the vanilla MRN. The model extensions are then compared to two other classifiers; Long-Short Term Memory and Support Vector Machine for an eye detection task and preliminary results shows that the extensions outperforms the two classifiers. These findings would be further explored to assess generalisation of these extensions and transferability to other domains. Future work will investigate the suitability of the MRN and its variants for complex higher-dimensional time series processing particularly applied for mortality prediction in the Intensive Care Unit (ICU) to identify points of deterioration in patient pathology and provide early warning indication of the deterioration and informing intervention measures.

REFERENCES

[1] M. Mahmud, M. S. Kaiser, A. Hussain, and S. Vassanelli, "Applications of deep learning and reinforcement learning to biological data," *IEEE transactions on neural networks and learning systems*, vol. 29, no. 6, pp. 2063–2079, 2018.

[2] H. M. Ali, M. S. Kaiser, and M. Mahmud, "Application of convolutional neural network in segmenting brain regions from mri data," in *International Conference on Brain Informatics*. Springer, 2019, pp. 136–146.

[3] S. W. Yahaya, A. Lotfi, and M. Mahmud, "A consensus novelty detection ensemble approach for anomaly detection in activities of daily living," *Applied Soft Computing*, vol. 83, p. 105613, 2019.

[4] M. B. T. Noor, N. Z. Zenia, M. S. Kaiser, M. Mahmud, and S. Al Mamun, "Detecting neurodegenerative disease from mri: A brief review on a deep learning perspective," in *International Conference on Brain Informatics*. Springer, 2019, pp. 115–125.

[5] O. Orojo, J. Tepper, T. M. McGinnity, and M. Mahmud, "A Multi-recurrent Network for Crude Oil Price Prediction," Proceedings of the IEEE Symposium Series on Computational Intelligence, Xiamen, China, Dec. 2019

[6] G. Rabby, S. Azad, M. Mahmud, K. Z. Zamli, and M. M. Rahman, "Teket: a tree-based unsupervised keyphrase extraction technique," *Cognitive Computation*, 2020, doi: 10.1007/s12559-019-09706-3, [epub ahead of print].

[7] D. Silver, A. Huang, C. J. Maddison, A. Guez, L. Sifre, G. Van Den Driessche, J. Schrittwieser, I. Antonoglou, V. Panneershelvam, M. Lanctot *et al.*, "Mastering the game of go with deep neural networks and tree search," *nature*, vol. 529, no. 7587, p. 484, 2016.

[8] S. Shalev-Shwartz and S. Ben-David, "Understanding machine learning: from theory to algorithms," New York, NY, USA: Cambridge University Press, 2014.

[9] T. O. Ayodele, "Types of Machine Learning Algorithms," in New Advances in Machine Learning, Y. Zhang, Ed.InTech, Feb. 2010. [Online]. Available: http://www.intechopen.com/books/new-advances-in-machine-learning/types-of-machine-learning-algorithms

[10] C. Ulbricht, "Multi-recurrent Networks for Traffic Forecasting," Proceedings of the National Conference on Artificial Intelligence, vol. 2, pp. 883–888, 1994.

[11] J. A. Tepper, M. S. Shertil, and H. M. Powell, "On the importance of sluggish state memory for learning long term dependency," Knowl. Based Syst., vol. 96, pp. 104–114, Mar. 2016.

[12] J. Binner, P. Tino, J. Tepper, R. Anderson, B. Jones, and G. Kendall, "Does money matter in inflation forecasting?" Physica A, vol. 389, no. 21, pp. 4793–4808, Nov. 2010.

[13] T. C. O'Connell, "Using Periodically Attentive Units to Extend the Temporal Capacity of Simple Recurrent Networks," PhD. dissertation,University at Albany, State University of New York, New York, NY,USA.

[14] A. Bouchachia and A. Ortner, "Self-learning recursive neural networks for structured data classification," in 2014 International Joint Conference on Neural Networks(IJCNN). Beijing, China: IEEE, Jul.2014, pp.808–815. [Online]. Available: http://ieeexplore.ieee.org/lpdocs/epic03/wrapper.htm?arnumber=6889804

[15] A. Hartstein and R. H. Koch, "A Self-Learning Neural Network," in Advances in Neural Information Processing Systems 1, D. S. Touretzky, Ed.Morgan-Kaufmann, 1989, pp. 769–776. [Online]. Available: http://papers.nips.cc/paper/189-a-self-learning-neural-network.pdf

[16] H.-W. Lee, N.-r. Kim, and J.-H. Lee, "Deep Neural Network Self-training Based on Unsupervised Learning and Dropout," IJFIS, vol. 17, no. 1, pp. 1–9, Mar. 2017. [Online]. Available: http://www.ijfis.org/journal/view.html? doi=10.5391/IJFIS.2017.17.1.1

[17] D. H. Nguyen and B. Widrow, "Neural Networks for Self-Learning Control Systems," 1990 p. 6.

[18] V. Reddy, "Self Trained Artificial Neural Network," 2004.

[19] A. Skabar, "Augmenting Supervised Neural Classifier Training Using a Corpus of Unlabeled Data," vol. 2479, 2002, pp. 174–185.

[20] A. Verikas, A. Gelzinis, and K. Malmqvist, "Using unlabelled data to train a multilayer perceptron, "Neural Processing Letters, vol. 14, pp.179–201, 12 2001

[21] R. Pascanu, T. Mikolov, and Y. Bengio, "On the difficulty of training Recurrent Neural Networks," arXiv:1211.5063 [cs], 2012.

[22] R. Brette, "Computing with Neural Synchrony," PLoS Comput Biol, vol. 8, no. 6, p.e1002561, Jun. 2012.

[23] J. Kirkpatrick, R. Pascanu, N. Rabinowitz, J. Veness, G. Desjardins, A. A.Rusu, K. Milan, J. Quan, T. Ramalho, A. Grabska-Barwinska, D. Hassabis, C. Clopath, D. Kumaran, and R. Hadsell, "Overcoming catastrophic forgetting in neural networks," arXiv:1612.00796 [cs, stat], Jan. 2017, arXiv:1612.00796.

[24] S. Sodhani, S. Chandar, and Y. Bengio, "Towards Training Recurrent Neural Networks for Lifelong Learning," arXiv:1811.07017 [cs, stat], Sep. 2019, arXiv: 1811.07017.

[25] P. Zhang, J. Xue, C. Lan, W. Zeng, Z. Gao, and N. Zheng, "Adding Attentiveness to the Neurons in Recurrent Neural Networks," in Computer Vision – ECCV 2018, V. Ferrari, M. Hebert, C. Sminchisescu, and Y. Weiss, Eds. Cham: Springer International Publishing, 2018, vol. 11213, pp. 136–152.

[26] W. Yin and H. Schütze, "Attentive Convolution: Equipping CNNs with RNN-style Attention Mechanisms,"Transactions of the Association for Computational Linguistics, vol. 6, pp. 687–702, Dec. 2018.

[27] W. Yin W. Yin, S. Ebert, and H. Schütze, "Attention-Based Convolutional Neural Network for Machine Comprehension," in Proceedings of the Workshop on Human-Computer Question Answering.San Diego, California: Association for Computational Linguistics, 2016, pp. 15–21.

[28] Z. Yang, D. Yang, C. Dyer, X. He, A. Smola, and E. Hovy, "Hierarchical Attention Networks for Document Classification," in Proceedings of the2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies. San Diego, California: Association for Computational Linguistics, 2016, pp. 1480–1489.

[29] L. Liu, R. Zhang, J. Peng, G. Li, B. Du, and L. Lin, "Attentive Crowd Flow Machines," arXiv:1809.00101 [cs, stat], Aug. 2018, arXiv: 1809.00101.

[30] F. Wang, M. Jiang, C. Qian, S. Yang, C. Li, H. Zhang, X. Wang, and X. Tang, "Residual Attention Network for Image Classification," in2017IEEE Conference on Computer Vision and Pattern Recognition (CVPR). Honolulu, HI: IEEE, Jul. 2017, pp. 6450–6458.

[31] Y. Kim, C. Denton, L. Hoang, and A. M. Rush, "Structured Attention Networks," p. 21, 2017 arXiv:1702.00887 .

[32] S. Chaudhari, G. Polatkan, R. Ramanath, and V. Mithal, "An Attentive Survey of Attention Models," arXiv:1904.02874 [cs, stat], Apr. 2019, arXiv:1904.02874.

[33] O. Rosler and D. Suendermann, "A First Step towards Eye State Prediction Using EEG," 2013, p. 4.

[34] F. Söylemez and B. Ergen, "Eye Location and Eye State Detection in Facial Images Using Circular Hough Transform," in Computer Information Systems and Industrial Management, K. Saeed, R. Chaki, A. Cortesi, and S. Wierzchoń, Eds. Berlin, Heidelberg: Springer Berlin Heidelberg, 2013, vol.8104, pp. 141–147.

[35] Ki Kim, Hyung Hong, Gi Nam, and Kang Park, "A Study of Deep CNN-Based Classification of Open and Closed Eyes Using a Visible Light Camera Sensor,"Sensors, vol. 17, no. 7, p. 1534, Jun. 2017.