Available online at www.sciencedirect.com

ScienceDirect

Procedia Computer Science 133 (2018) 298–305

Procedia Computer Science

www.elsevier.com/locate/procedia

International Conference on Robotics and Smart Manufacturing (RoSMa2018)

Differed Restructuring of Neural Connectome Using Evolutionary Neurodynamic Algorithm for Improved M2M Online Learning

Ankush Rai[a]* , R. Jagadeesh.Kannan.[b]

[a,b] *School of Computing Science & Engineering, VIT University, Chennai*

Abstract

Ontological nodes and training material, in any case, contain however a little part of the quickly developing volume of unstructured contents accessible across training datasets ranging from the greater part of heterogeneous "learning". Recovery of these assets is constrained to keyword search, which is in a far-reaching way incognizant in regards to the normal sequential semantic connection in which the query is often requested and all the more essentially for more importantly for development of a perception about a learned concept through the sequential arrangement of information. Numerous core topics covered in several referential sources when no two are precisely similar, with one frequently more suitable when seen in a connection of an ontological relational modalities of semantic web and online training of network based on continuous streaming database; while other on giving major insights of practical implementation. The objective of this work is to present an evolutionary neurodynamic framework to mine such process based learning validated by the visual recording & extraction of neural weights through neurodynamic neural experiences when going through the referenced content. The motivation of this behavioral modeling of autonomous machine's reaction working as the feedback of the learning content set forth above hinges on the principal capacity to gather a structure of a curricular graph automatically from the training datasets. As it empowers the acknowledgment of: 1) Sharing of neural experiences with the help of structurally sequenced networked topology of such large sample of training datasets resources - with the idea of dynamic analysis and linking of key resources in the mechanism with which the learner develops a general concept about the subject which can be latter re-used or shared with other user's and thus facilitating the qualitative evaluation of the ontological nodes content and its contribution in facilitating deep grasp of the data from feedback loop. 2) Obtaining information driven ontological knowledge base from the curricular relations inside and crosswise over other learning resources and 3) Generating customized, focused curricula, utilizing the tremendous assets of heterogeneous modalities on the training datasets.

*Keywords:*Neural Modelling; Online Learning; M2M Learning; Self-Modelling.

* Corresponding author. Tel.: +91-940609251; fax: +0-000-000-0000 .
E-mail address: ankushressci@gmail.com

1877-0509
10.1016/j.procs.2018.07.037

1. Main text

The past decade has witnessed the emergence of kernel based online learning or simply online learning paradigm which is just a subset of machine learning with a combination of ubiquitously computational load balancing approach over a network [1-3,5]. This online algorithms tend to work in closed loops or rounds, where each of this round generates several new instances of confusion matrices for the consolidate sets known by the name as support set. Now, since all confusion matrices and prediction states isn't sufficient to be stored in the support sets which lead to the problem of relative memory loss bounds. The learned data in an online algorithm is known as hypothesis; thus due to this relative loss in memory bound for the enough support sets to exist required for decision making for online computational jobs is hindered. Therefore, there are several functions are deployed in an online leaning approach, wherein the classification of the weighted sum of combination are stored in support sets. Hence, making the system to update itself with the variation of the weighted sum and prioritize or predict accordingly with the modified online hypothesis. But since, if this continues so on then the support required to keep the online hypothesis will grow unbounded and leading to memory explosions. Thus, far this has been a major problem in online learning algorithms and is required other several novel methods for its resurrection and effective applicability.

Since, such algorithm are associated with autonomous web agents, big data analytics and real-time data analysis. Therefore a solution to this problem will give boost to this research scenario. To accomplish the same the researchers in the past have used a method which relies on the term known as budget. The budget is the limit of the support set which uses heuristic method to decide which instance has to be kept and removed [2,3,6]. Others strategy include the use of NORMA & SILK which are very similar on such an approach [5,7]. In an another study, they have used Forgetron algorithm, whch relies on an hybrid approach of memory budget and relative mistake bounds [8]. Latter in the studies a stochastic algorithm is presented which gives similar performance range on an average basis [9,11]. The Gaussian approach is another approach used for the same where instances are discarded and spanned over the mistakes in support sets [12]. Due to it's effectively of avoiding hinge loss its gives sparser solutions but not as much as compared with the Langford. Here the few parameters are proposed to induce scarcity in online learning budgets [13].

Nomenclature

m & n	Edge & vertices of the given data frame
$r_{i,j}, g_{i,j}$	Streaming data feedback intensities at instances i,j.
$CCN(N)$	Pool of feature set entries and coded into feature set networks
M	Represents the color model of the given dataset
$\gamma_M \& \sum_M$	Mean & covariance of the feature distribution
Y_{edge}	Ontological patches from sampled block set $B_{\gamma M}$
$\|\|.\|\|_F$	Represents frobenius norm,
D	Dictionary of all blocks $D = [d1, d2, \dots dn]$
X	Sparse matrix $X = [x1, x2, \dots xn]$.
f	Set of categorized streaming feedback band
P	Uniformity predicate defined over groups of connected streaming feedback.
R	Mean value; $g_i \& g_{i+1}$ are the lower bound and upper bound respectively
S	Splitting function
$f(R_i, L_i)$	Function represented is recursively with $R_i \& L_i$
t_{f_t}	Segmented region with integrated neural features.
T	*Time*
a_t	Action
s_t	State
r_t	Reward
X(i, p)	Indicator to the event that the solution is in state *i* during the p_{th} phase of feature instance
N_i the number of phases of state *i*.	
L_N	Universal set of level for the telematic microservice actions

p_i & p_2	Adjoint sequence pairs
S_{p_1,p_2} & S'_{p_1,p_2}	Sets of sequence density constraint
t_i	Collection of patterns
f_1, f_2	Two delay frames with a minimal time delayst_i
AS_i	Automated classified action sets.
C_{tar}	Desired target output
C_{out}	Actual network output.
w_{2r1}	Weight of the connection

2. Methodology

Exact localization of a neural component assumes an essential part in feature sets, and differed restructuring. However in genuine applications, on account of the distinction fit as a fiddle and the nature of the ontological edges, it is hard to find the neural element accurately. There are numerous routines accessible for neural element extraction, regardless of this, neural component extraction for feedback acknowledgment is still a daunting issue. The issue of extricating the element of neural population data is comparable to the issue of feature division from certain feature zones. There are two sorts of division strategy: model-based division and feature based division. Feature based division techniques, for example, the "Live-wire system", are more exact in highlight limit outline, however they are poor in limit acknowledgment, for example, highlight locale extraction by coordinating article zones in the aggregate feature. Thusly, we introduce the accompanying novel way to deal with focus the ontological edges of the neural elements. The proposed framework uses the following algorithm which ensures the carryover tracking and representation of the accurate edge maps continued across the several data frames of the given neural sequences. The algorithm fills those segments with networks of feature sets and hence ideally describes perfectly the segment boundaries in dynamic environment. The idea here is to normalize the given neural scheme, thereupon the feature sets are segmented and finally pooled in together to form a network of feature sets which is carried over to the next data frames and so on. The algorithm for the same is given as:

Algorithm: Network Coded Feature set Algorithm (NCCA)

Input: Data sequence with N frames, each frame is characterized with $m \times n$ ontological nodes, where m & n are the edge & vertices of the given data frame, $r_{i,j}, g_{i,j}$ are the streaming data feedback intensities at instances i,j.

Output: Coded Feature set Networks CCN.

Step 1: Segment block patches:

for$r_{i,j}$ *to m'*

for$g_{i,j}$ *to n'*

if$(B_{\gamma_M} \leq \sum_M P(r_{i,j}, g_{i,j}))$

{

$Y_{edges} = P(edges|rg, \underbrace{min}_{DX} \|Y - DX\|_F^{\gamma_M})$ //for creating sparse matrix

else

$Y_{edges} = P(edges|rg, \underbrace{min}_{DX} \|DX\|_F^{\gamma_M})$ //for Matching Feature sets

}

Step 3: Create pooling of feature set entries and coded into feature set networks

$$CCN(N) = \left[\sum_i^N Y_{edges} \sum_i^N X - \sum_i^N \underbrace{min}_{DX} \|DX\|_F^{\gamma_M}\right] \quad (1)$$

Step 4: End Process.

Where, the value M represents the color model of the given dataset, which is embarked as low feedback intensity after the pre-processing. γ_M & $\sum_M$ are the mean & covariance of the feature distribution based on feedback intensities in *rg* feedback scheme scheme after pre-processing.$Y_{edge} = [y1, y2, \dots yn]$ are the ontological patches

from sampled block set B_{γ_M} , $\|.\|_F$ represents Frobenius norm, D is the dictionary of all blocks $D = [d1, d2, \dots dn]$with size of 5x5 streaming feedback and X is the sparse matrix $X = [x1, x2, \dots xn]$. Here X is used to couple the several of the pooled feature sets .

This avails a knowledge guidance as an imperative hypothesis for the consequent algorithms to easily extract the neural features from the above derived output with mostly even and continuous feedback intensity values of the ontological edges and geometry of feature base topological connections. Through examination of the attributes of coordinate distributions on ontological database in long haul neural differed restructuring tests, in view of weighted spatial limitation of neighboring streaming feedback the edge estimation of the pooled feature sets is resolved through N. Now, we need to extract and segment the neural feature from the simplified feature set networks. Thereby, the output frame so derived is a two dimensional (2 D) function, *f*(x, *y*), where *x* and *y* are the coordinate values in spatial domain; and the magnitude of *f(x, y)* is the non-varying continuous intensity value of streaming feedback at (x, y). If x, y and the magnitude of *f(x, y)* are discrete quantities. Neurodynamic ensemble may be represented as two dimensional matrices whose elements are intensities of streaming feedback present in neurodynamic ensemble. Almost all neurodynamic ensemble processing related operations operate on this streaming feedback either in spatial domain or in frequency domain or transform domain. The function *f(x, y)* can be expressed as:

$$f(x,y) = \begin{bmatrix} f(0,0) & \cdots & f(0,N_y-1) \\ \vdots & \ddots & \vdots \\ f(N_x-1,0) & \cdots & f(N_x-1,N_y-1) \end{bmatrix} \quad (2.1)$$

Now, each digital neurodynamic ensemble has certain finite number of elements characterized by some coordinate values and feedback intensity value. The coordinate indicates the position of streaming feedback in an neurodynamic ensemble. In Equation (1) the neurodynamic ensemble elements $f(N_x-1, N_y-1)$ represent the maximum number of resolution starting from *f (0,0).* Suppose that 'f' is the set of categorized streaming feedback band and 'P' is a uniformity predicate defined over groups of connected streaming feedback. Segmentation is simply a partitioning of the set F into a set of connected feature points (P_1 , P_2, ….P_n) such that $\bigcup_{i=1}^{n} P_i = F$ with $P_i \cap P_j = \emptyset when i \neq j$. The uniformity predicate i = 1 streaming feedback represented as P (P_i) is true for all regions $P_i \& P(P_i \cup P_j)$ and is false when P_i is adjacent to P_j.. The thresholding algorithm for binary neurodynamic ensembles is applied as:

$$f_t := \sum_{i=0}^{m} r\ (\{g_i \leq f_{block} < g_{i+1}\} \quad (2.2)$$

Where, r( ) is the mean value; $g_i \& g_{i+1}$ are the lower bound and upper bound respectively of the given thresholding streaming feedback boundary condition. The unnatural bias for partitioning is avoided by selecting small sets of points and different measure of dissociation. The problem with such criterion for thresholding is that it does not consider association with clusters. In order to circumvent this problem, the cost of thresholding at runtime as a function of the total streaming feedback threshold to all those levels formed in the above step is determined and taken in account through the streaming feedback association rule.

Thus, we have the generic equation normalization is defined as:

$$f_t(N_x, N_y) = \sum_{i=0}^{x} r\left(\frac{cut\ (N_x-1, V)}{assoc\ (N_x-1, N_y)} + \frac{cut\ (N_x, N_y)}{assoc\ (N_y, V)}\right) \quad (3)$$

Where, assoc (N_x,N_y) = $\sum_{x \in m, v \in n} W(u, v)$ is the total connection from streaming feedback of set A to all set B. By using this definition of the disassociation between the groups, small isolated points in ontological database are partitioned out and will no longer have distinct *N*values, since the cut value will almost be a large percentage of the total connections from the small set to all other streaming feedback. If no other level changes are found then terminate the operation. The mechanism of segmented neurodynamic ensemble is finally generated after extraction operation.

Thus, let us suppose that levels based dependencies between different neural parts can be expressed as *p(A|B)* where A is the sets of nodes estimated in previous steps during normalization and B is the voting element for A which express the neural description for the local sub patches in form of a sets of nodes given by the training sets of a number of neurodynamic ensembles on iteration basis. Thus, the location of different parts is dependent i.e.,

$$p(A|B) = \prod_{i=1}^{N} p(f_t|B) \tag{4}$$

But since, above equation creates a consolidated regions R_i from Levels L_i and Streaming feedback p_i. Therefore, the entropy H of two positions can be mathematically defined as:

$$H(B) = \sum_{n=1}^{N} \frac{\sum_{i \in R_i}(A|B)}{\|B\|} \tag{5}$$

$$t_{f_t}(R_i, L_i) = \begin{cases} H(B), if \;\; f_t(R_i, L_i) < S \\ H(A \cap B), if \;\; f_t(R_i, L_i) > S \end{cases} \tag{6}$$

$$S = \frac{t_{f_t}}{\nabla t_{f_{t+1}}} \tag{7}$$

Where, S is the splitting function and $f(R_i, L_i)$ which is the function represented is recursively with R_i&L_i as its hierarchical input parameters and t_{f_t} is the segmented region with integrated neural features. Now for modeling of neurodynamic evolution with several categorization based on nature of compliance and non-compliance with the content of the ontological nodes which is classified into positive and negative. Following the above step the generated data need be forwarded to semantically filter out optimal policy (i.e, correlated state action pair) with higher reward through the help of following distributed reinforcement learning. A policy P is memory-less technique, i.e., it primarily depends only upon the current state and not onto its history. Thus, a deterministic strategy P assigns each state a unique action. While taking after a strategy P we perform at time t action a_t at state s_t and observe a reward r_t (distributed according to $R_{MDP}(s, a)$). and the next state s_{t+1} (dispersed according to $P^{MDP}_{S_t, S_{t+1}}(a_t)$). We consolidate the sequences of rewards to a single value called the return, and our goal is to maximize it. This gives us linear time complexity for the synchronous learning rate. Where, symmetry breakdown allows us to ease the problem of extracting semantic rule by looking for the inter-correlation between symmetry of the state pairs and the symmetry. Hence, the relationship between it can be learned in one shot for rule generation, which is given as:

$$x(i,p) \leftarrow \left(\sum_{t=t(i,j)+1}^{t(i,j+1)-1} F_j(t) \right) - z(i,j) - w(i) \tag{8}$$

Here, *x(i, p)* be an indicator to the event that the solution is in state *i* during the p_{th} phase of feature instance and n_i be the number of phases of state *i*. Thus, forming a dynamic sequence. The nodal degree distribution was fat-tailed with high-degree hub nodes to be located in the above mentioned excitatory neural network using sequence of information to excite the necessary regions and asses the information in an associative form. This enables several services all at once to not only learn but it enables it to embark the cross relationship between various data for prediction or simulation based logical conclusion; herein the processing is done over neural net based shell environment. Computationally, this topology was embedded parsimoniously, in terms of the connection distance between co-activated nodes. Most connections or ontological edges were separated by short sequence of excitatory data, significantly shorter than random networks; the parallel policy based learning equation is given as based on Instance of Window's Workspace W (b), Instance of machine's end U and the filtered Action Sets is given by AS_i with Matrix Model of Tree of Actions M_x. Compute the Pointing Correlation state P as:

$$P = \frac{1}{L_N} \sum_{p_i}^{L_W - 1} \left[\sum_{p_2}^{L_U - 1} S_{p_1,p_2}(t_i, f_1, f_2) \right] \left[\sum_{p_2}^{L_U - 1} S'_{p_1,p_2}(t_i, f_1, f_2) \right] \tag{9}$$

Where, L_N are the universal set of level for the telematic microservice actions, p_i & p_2 are the adjoint sequence pairs with the levels L_W & L_U respectively, S_{p_1,p_2} & S'_{p_1,p_2} are the sets of sequence density constraint layout for the action sets positioning with its patterning saved in levels and between its intersection of adjoint pairs and the

superpositioned pair density layout of differing state at the service's instance of the frame U. Also, t_i is the collection of patterns for the weighted superposed state P_c(initially its value is set to 0), f_1, f_2 are the two delay frames with a minimal time delayst_i [11-13]. Thus, we calculate the Tree of Action based on continuous feedback loop:

$$M_x = \begin{pmatrix} t_1 \begin{bmatrix} P_1 \\ P_4 \\ P_8 \end{bmatrix} = AS_1 \\ t_2 \begin{bmatrix} P_3 \\ P_9 \\ P_6 \end{bmatrix} = AS_2 \\ t_3 \begin{bmatrix} P_2 \\ P_5 \\ P_7 \end{bmatrix} = AS_3 \\ \vdots \\ t_i \begin{bmatrix} P_0 \\ P_5 \\ P_c \end{bmatrix} = AS_i \end{pmatrix} \quad (10)$$

Where, AS_i is the automated classified action sets. Again, to optimize the above derived sequence of blocks we use membrane computing to carter distributed services with parallel policy based learning from several agents. Here, C_{tar} is the desired target output and $C_{out}$is the actual network output. The value of C_{out} is determined as: $C_{out} = [Q_2^{(1)} Q_2^{(2)} \ldots . Q_2^{(N)}]$ where $Q_2^{(1)}, Q_2^{(2)}, \ldots, Q_2^{(N)}$ are the network outputs of each agent using policy based learning. The individual network outputs can be computed as:

$$Q_2^{(1)} = \sum_{r=1}^{N_g} w_{2r1} Q_1(r) \quad (11)$$

$$Q_1(r) = \frac{1}{1 + exp(-w_{1r1}. C_{in})} \quad (12)$$

Where w_{2r1} is the weight of the connection from the $2r_{th}$ input element to the 1_{th} hidden unit. The above equation is a distributed function of several intermittent output layer and hidden layer respectively. Adjusting the weights of all neurons by $w = w + \Delta w$, where Δw is the change in weight estimated as: $\Delta w = \gamma . Y_2 . BP_{err}$, where γ is the learning rate. Generally the value of learning rate is between 0.2 to 0.5.

3. Results and Discussion

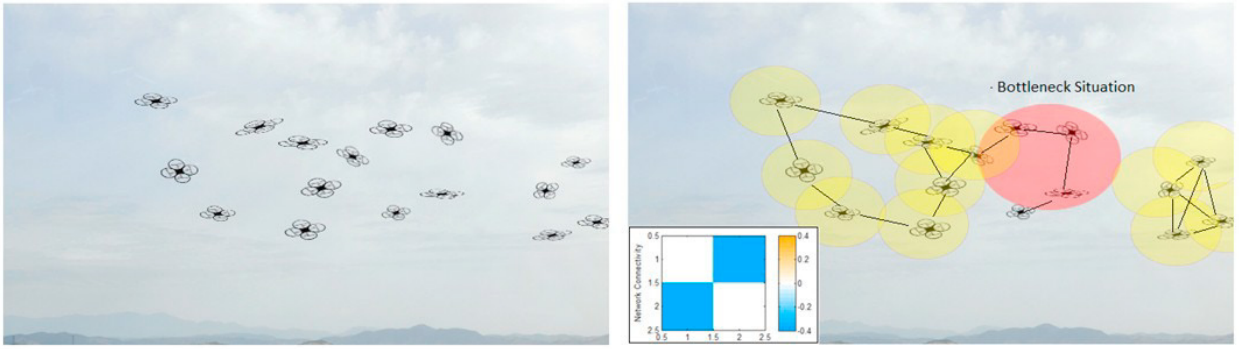

Figure 1: (a) Snapshot of the experimental multi drone setup using the proposed co-simulated organization of cyber physical network system (b) Bottleneck state of multi drone network where yellow aura represents drones in connection and black lines represent routing links, red aura denote the lost links between drones. Inset is the graph of connectivity strength.

Note that in figure 1 represents the typical output for the reconfiguration of neurodynamic evolution in certain amount of drone communication and flight timeline which appears to be performing superior when there are events showcasing micro neural experiences within the time range of 30 ms. One possible explanation for this the neural cues includes the the time evolution steps required to 3D tetrahedron to come to its equilibrium takes longer time in processing; of which in this context takes 2~2.7 ms in processing it back to equilibrium for next consecutive frame. Furthermore, because such activity is normally followed by the other complex flight movements, instead of neutral neural experiences during the flight, which makes this geometric topology generally, lags to follow the neurodynamic evolution due to its time required for its restoration back to equilibrium after a complex evolution to match with the orientation and positioning of drones and its neural features. Such events are categorized as post hoc. Though, the lag isn't considered to be that huge to miss the general and important micro neural experience to assess the learning neural experience into positive or negative. Regardless of this critical expect of this findings is that we are successfully capable to build a system which recognizes and asses the sub parts of dynamic learning process into positive or negative. We are able to build up a 3D networked topology of the ontological resources that are used by the subject in this case drones in the autonomous flight experiment.

This networked topology of the learning training datasets resource about the specified subject area works as an neural experience network to be able to share with the other subjects for better and efficient learning by simply turning feedback based streaming training datasets as neural experience sharing modules between several drone u units, an example of this networked topology used during the course of study is shown in the following figure 4. In the study the most prominent of this networked topology in regards of the ontological nodes content is shared among the individuals to ask for their changed view towards readily available sequenced learning resources required to build a comprehensive concept about the ontological nodes content. This immensely improved self-efficacy of the drone units and aroused more interest towards binding of neural mechanism with ontological database to reduce the failure in online learning. The following table enlists the comparison of the neurodynamic evolution recognition performance between the proposed framework and that of the previously known methods.

The results summarized in table 1 are obtained using SVM (RBF), SVM (Poly), Naive Bayes, K-NN (k=3) and the proposed framework for 10-fold cross validation technique& also by increasing the training sets to *80%*. As can be inferred from the above mentioned table the overall performance of the proposed method peaks to 93.8% from a total number of $400 \times 300 \times 30$ instances with less computational cost in both training & testing with the other competing algorithms; out of which 1440 instances are misclassified due to the lag already discussed above. Also, there isn't any ambiguity is noticed with the recognition system to confused with the neurodynamic evolution which were the no so true case for the competing algorithm, as such algorithms are bound to misclassify happiness with disgust, sadness with neutrality owing to the similarity between such features thus relying over match score level which is just another add up in more computational time. Thus, evaluating the current scenario the proposed framework seems to be efficient in neurodynamic evolution and fit suitably in online learning environment for cyber physical systems with a little lag and additive feature of feedback tracking due to its geometrical nature.

Table 1.Comparisison of accuracy achieved in neurodynamic evolution recognition.

Methods	Accuracy (%)
10 fold cross validation	
SVM (RBF)	90.3
SVM (Poly)	88.8
Naive Bayes	68.14
Proposed Method	92.8
80% training, 20% testing	
SVM (RBF)	86.7
SVM (Poly)	85.2
Naive Bayes	70.3
K-NN (k=3)	83.3
Proposed Method	94.8

4. Conclusion

The issue of getting deferred neural experiences amid the learning process in training datasets learning situations is not just an issue for analysts. The presented framework offers a methodological possibility is to make training datasets online learning for streaming database more efficient, measurable, and add more data drive circumstances for more research in the field. It allows us to build an on the fly training datasets based semantic as per the autonomous machine's level of understanding with continuous recording of deferred variances to suitably modify the learning process. The learning neural experience recorded in form of networked topology of semantic features gives the training datasets to share more of their neural experiences with each other within the swarms or as the swarm size increases Advancement introduced in this paper will render conventional learning into more of process based learning. As noted the autonomous machine's learning was more synchronous with the learner. This study will come in handy when training distributed modules of cyber physical systems.

References

[1] Cho, Y. and Saul, L. (2010). Large-Margin Classification in Infinite Neural Networks. Neural Computation, 22(10), pp.2678-2697.

[2] Louche, U. and Ralaivola, L. (2015). Unconfused ultraconservative multiclass algorithms. Machine Learning, 99(2), pp.327-351.

[3] Burkhardt, S. and Kramer, S. (2017). Online multi-label dependency topic models for text classification. Machine Learning, 107(5), pp.859-886.

[4] Jorge, J. and Paredes, R. (2018). Passive-Aggressive online learning with nonlinear embeddings. Pattern Recognition, 79, pp.162-171.

[5] Scardapane, S., Comminiello, D., Scarpiniti, M. and Uncini, A. (2015). Online Sequential Extreme Learning Machine With Kernels. IEEE Transactions on Neural Networks and Learning Systems, 26(9), pp.2214-2220.

[6] Even, G., Halldórsson, M., Kaplan, L. and Ron, D. (2008). Scheduling with conflicts: online and offline algorithms. Journal of Scheduling, 12(2), pp.199-224.

[7] Scardapane, S., Comminiello, D., Scarpiniti, M. and Uncini, A. (2015). Online Sequential Extreme Learning Machine With Kernels. IEEE Transactions on Neural Networks and Learning Systems, 26(9), pp.2214-2220.

[8] Dekel, O., Shalev-Shwartz, S. and Singer, Y. (2008). The Forgetron: A Kernel-Based Perceptron on a Budget. SIAM Journal on Computing, 37(5), pp.1342-1372.

[9] Cesa-Bianchi, N., Conconi, A. and Gentile, C. (2004). On the Generalization Ability of On-Line Learning Algorithms. IEEE Transactions on Information Theory, 50(9), pp.2050-2057.

[10] Cesa-Bianchi, N., Conconi, A. and Gentile, C. (2005). A Second-Order Perceptron Algorithm. SIAM Journal on Computing, 34(3), pp.640-668.

[11] Cavallanti, G., Cesa-Bianchi, N. and Gentile, C. (2007). Tracking the best hyperplane with a simple budget Perceptron. Machine Learning, 69(2-3), pp.143-167.

[12] Csató, L. and Opper, M. (2002). Sparse On-Line Gaussian Processes. Neural Computation, 14(3), pp.641-668.

[13] He, L. and Wang, Y. (2018). Image smoothing via truncated ℓ 0 gradient regularisation. IET Image Processing, 12(2), pp.226-234.