2014 International Joint Conference on Neural Networks (IJCNN)
July 6-11, 2014, Beijing, China

Extending Dynamic SOMs to Capture Incremental Changes in Data

K.M.T.V. Ganegedara, L.C. Vidana Pathiranage, U.A.R.R. Gunarathna, B.S. Wijeweera and A.S. Perera
Department of Computer Science and Engineering
University of Moratuwa
Moratuwa, Sri Lanka
{thushang.09, lasindu.09, ruwan.09, buddhima.09 and shehan}@cse.mrt.ac.lk

D. Alahakoon
School of Information and Business Analytics
Deakin University
Melbourne, Australia
d.alahakoon@deakin.edu.au

Abstract— **Humans learn in an incremental manner. Due to this reason, humans continuously refine their knowledge of the environment with the experience gained. Many strides have been made in the machine learning area to exploit the power of incremental learning. Incremental learning, in contrast to one-time learning is far more useful and effective when data is not completely available at once. Here, we investigate an unsupervised incremental learning algorithm known as Incremental Knowledge Acquisition and Self Learning (IKASL) algorithm. IKASL algorithm is able to capture knowledge in an incremental manner, without disrupting past knowledge. Furthermore, IKASL algorithm encodes acquired knowledge in such a way that it can be used to acquire new knowledge more efficiently. This paper discusses several limitations of original IKASL algorithm and proposes several extensions to the original algorithm which enhances its performance. These modifications include influencing spread factor, implementing fuzzy integral based generalizing technique, etc. Furthermore, the paper report observations of several experiments conducted with several datasets to assess the necessity and value of incremental learning in the real world. These experiments are carefully designed to reflect interesting characteristics of the IKASL algorithm.**

Keywords—incremental knowledge acquisition; neural networks; dynamic som; visual semantic patterns;

I. INTRODUCTION

During the early stages of Machine Learning, it often has been an implicit assumption that, a complete training data set is available at the execution time of the algorithm. But far more interesting results could be achieved if the algorithm encompasses the ability to learn over a period of time, incrementally, in contrast to one-shot learning. Furthermore, incremental learning prevents the need of "re-inventing the wheel" every time new information arrives. Moreover, incremental learning becomes essential in the learning process. Christophe Giraud-Carrier suggests that, in learning grammar, recurrent networks fail to learn when inputs are provided all at once, but succeeds when presented incrementally (e.g. easy-to-hard) [1].

Furthermore, humans learn in an incremental manner. Initially, a human will formulate a basic understanding with available information and continue to refine the understanding as new information is acquired. Throughout the entire life, a person will learn in various ways such as acquiring knowledge, observations and experience.

Inspired by both necessity to overcome the limitations of one-shot learning and the humans' ability to learn incrementally, a manifold of incremental learning algorithms have emerged over the last decade.

Incremental Knowledge Acquisition and Self Learning (IKASL) algorithm [2], can be recognized as a novel incremental learning algorithm. IKASL algorithm is capable of acquiring knowledge in an incremental manner, while preserving the past knowledge. IKASL algorithm can be understood as a n-layer dynamic self-organizing feature maps. Due to the novelty of this algorithm, it has not been fully explored for its functionality and usability yet. In this paper we are proposing several extensions to the original IKASL algorithm which enhances the performance of the algorithm. In addition, the paper focuses on evaluating the value of incremental learning and its real world applicability by testing IKASL algorithm against several datasets.

This paper is organized as follows. Section 2 provides the background knowledge of Self-Organizing Maps (SOM) algorithm and Growing Self-Organizing Maps (GSOM) algorithm. Section 3 outlines the IKASL algorithm and proposes several improvements to the original algorithm which enhances performance. Section 4 presents several experimental results obtained by applying IKASL algorithm to several real-world datasets. Section 5 concludes the paper.

II. BACKGROUND STUDY

Before delving into technical details of the study, it is important to investigate existing incremental learning algorithms and their limitations.

A. Other Incremental Learning Techniques

Here, two neural network based incremental learning algorithms, one non-neural-network based incremental learn-

978-1-4799-1484-5/14/$31.00 ©2014 IEEE

ing algorithm and the limitations of those algorithms are discussed.

The lifelong learning cell structures (LLCS) [3] algorithm is an extension to the Growing Cell Structure algorithm (GCS). GCS is based on the SOM algorithm but the conventional two-dimensional grid of the SOM has been replaced by a network of nodes whose connectivity defines a system of triangles. Algorithm uses heuristics to both add and remove network nodes and connections. The algorithm can locally adapt the feature map for node insertions and for learning by heuristics. The LLCS algorithm focuses on local adaptations for incremental learning while ignoring correlative learning of the self-organizing process.

The SOINN [4] algorithm consists of a two layer network. The training results of the first layer are used as the training set for the second layer. The two locally accumulated error based insertion schemes, between class insertion and within-class insertion allows the SOINN to incrementally learn patterns in input data. The high node insertion requirement of the SOINN network and use of the utility parameter, constrains topographical organization of the data space input to the SOINN network.

Though, the popular density-based clustering method OPTICS [5] is not suitable for data streams, a variant called OpticsStream [6] can be used to visualize the clustering structure and structure changes in data streams. The results produced by the OpticsStream algorithm allow the user to observe the dynamic nature of cluster structure (i.e. Changes in cluster structure). Even though OpticsStream, provides intuitive representations of the clustering structure as well as how the results changes through time, it tends to find clusters of data points based on some notion of proximity, but completely ignore the temporal aspect of the data stream which can be crucial to studying patterns in data in detail.

IKASL algorithm attempts to overcome the limitations explained in the above incremental learning techniques. The next sections describe the algorithms which forms the basis for IKASL algorithm.

B. *Self-Organizing Maps*

Analyzing high-dimensional and complex data sets is an intricate task. Due to this reason, dimensionality reduction techniques can be very useful. Clustering techniques can be used to scrutinize high-dimensional data with less effort. Among various clustering techniques Self-Organizing Map (SOM) [7] plays a major role. SOM is popular because it is an unsupervised learning algorithm inspired by the Hebbian learning rule and the topology preservation property. Moreover, results of a SOM algorithm can be viewed in a 2D map for visual analysis.

C. *Growing Self-Organizing Maps (GSOM)*

Growing self-organizing map (GSOM) [8] is an extension of SOM algorithm. GSOM algorithm overcomes several limitations of SOM. GSOM algorithm has the ability to grow the feature map, if the neural network is insufficient to represent the input space. Unconstrained learning and node growth in the GSOM as opposed to constrained learning in the SOM will ensure the development of a natural topology of the data space, supporting incremental learning. Also, due to its dynamic structure, the GSOM achieves the same amount of spread with lesser number of nodes, and as such will provide a useful advantage in mapping large data sets [8]. In addition, such flexible structure provides a better visualization of the clusters and outliers in the data.

GSOM is initialized randomly with four nodes. As initial four nodes are organized in a square shape; it allows the map to grow in any direction depending on the input values. Moreover, this organization is a good starting position to implement a 2-D lattice structure [8]. Then the inputs are fed to the network. If a node's error value exceeds a parameter known as Growth Threshold (GT), the network is allowed to grow from the boundaries to represent the input space better.

III. EXTENDED IKASL ALGORITHM

A. *Original IKASL Algorithm*

IKASL algorithm [2] is a Hebbian rule based incrementally learning algorithm which is both stable and plastic. IKASL algorithm can be perceived as a n-layer structure, where n is the number of learning periods. Each layer comprises two sub-layers; a learning layer and a generalized layer. Learning layer embodies the GSOM functionality and generalized layer encodes a generalized representation of the learning layer immediately below. It is important to note that, the layers in IKASL are virtual and will come to existence as required by the learning process. Structure of the IKASL outcome is depicted below in Fig. 1.

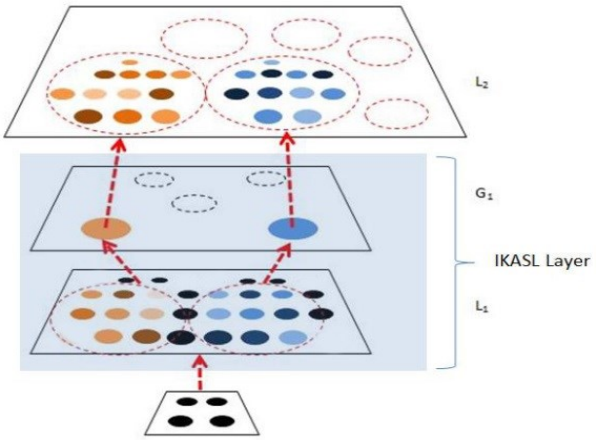

Fig. 1. Output of IKASL Algorithm

$L_0,L_1,L_2,\ldots,L_n$ - Learning layers and $G_0,G_1,G_2,\ldots,G_n$ - Generalized layers

The first learning layer, which marks the inception of the learning process, starts with a 4 randomly initialized nodes. Thereafter each dataset is fed to the feature map comprising of 4 nodes. Then, for each tuple in the dataset, a winner node will be selected from the nodes based on selected distance measure (e.g. Euclidean distance). Then, the weights of the winning node will be adapted to reflect the input. Moreover,

feature map will grow if it is insufficient to represent the input space. This process will take place iteratively for all the tuples in the dataset for a predefined number of epochs.

After the learning phase, the generalization phase transpires. In the generalization phase, winning nodes from the previous learning layer are identified. Next, these identified nodes need to be generalized in such a way that, the result should reflect two important outcomes of the learning process; the knowledge embodied in the weight vector of the winning node (primary outcome) and the knowledge of winner's neighborhood nodes (secondary outcome) [2].

The generalization could be achieved by employing fuzzy integral. The resulting new nodes (Generalized nodes), form the generalized layer.

In general, Ln learning layer will produce Gn generalized layer after the generalization phase. Then, next input set will be fed to the Gn layer. Therefore, Gn will form the basis for the learning phase which will take place in Ln+1. Furthermore, it is important to note that, each individual node in the generalized layer will grow into individual feature maps.

Following subsections delineates the modifications proposed to enhance the performance of the original IKASL algorithm.

B. New Disparity Measure

After generalizing the layer Ln, Gn will come into existence. Then, the next input sequence is fed to Gn which will form the next learning layer, Ln+1. While feeding the input sequence it is possible that input(s) exist which are substantially dispersed from the nodes in Gn layer (i.e. outliers). In order to identify such outliers, a disparity threshold (DT) has been introduced. If the distance between the winner node and input(s) exceed DT, it implies that the input is an outlier with respect to the node. Therefore, in order not to forget the existing knowledge, if any input(s) exceed DT, the winner node will remain unchanged, rather than overwriting current knowledge of the node with the new knowledge. DT is calculated as follows,

1) Consider an inputs sequence D = d_1,d_2, ..., d_m with d_i=(x_{i1}, x_{i2}, ..., x_{ip})

2) Calculate the standard deviation for each column j

$$\sigma = \sqrt{\frac{\sum_{i=1}^{m}(x_{ij}-\mu)^2}{|D|}} \quad (1)$$

3) Define, 'k' as the number of columns with a standard deviation > 0.25

4) Finally, the disparity measure DT is calculated as,

$$DT = \sqrt{0.5^2 \times k} \quad (2)$$

C. Increasing Spread Factor for 1st Learning Cycle

The concept of spread factor originates from the GSOM algorithm. Spread factor determines the spread of a feature map. In order to obtain a good spread in the GSOM algorithm, spread factor needs to be predefined by domain experts. In the IKASL algorithm the first layer forms the foundation for subsequent learning. Therefore, if spread is low initially, it will affect negatively for subsequent layers. Therefore, it is important to have a substantial spread in the first layer, even if the user specifies a low spread factor. We propose (3) to increase the low spread factor. Fig. 3 presents how the spread factor is affected by the proposed modification.

$$SF \times \left(1 + \left(\frac{1.2}{\sigma\sqrt{2\pi}} \times e^{-\frac{1}{2}\left(\frac{SF}{\sigma}\right)^2}\right)\right) \quad (3)$$

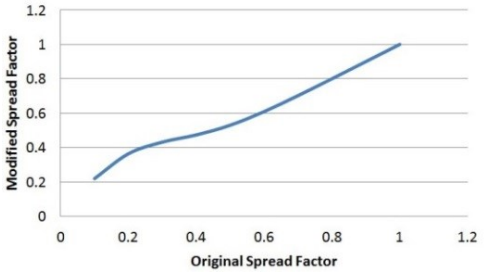

Fig. 2. Graph showing how spread factor is affected by the proposed function (with σ=0.2)

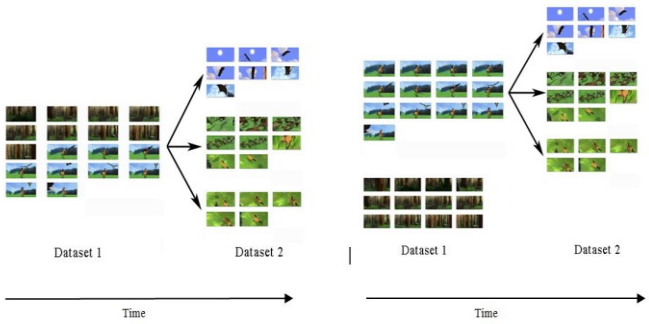

Fig. 3. Left hand side shows the clusters with SF = 0.3 without SF Modification. Right hand side shows the clusters with SF = 0.3 with SF Modification. As it can be seen due to the enhancement of spread factor, right side result has produced better clusters than the left side

D. Integrating Fuzzy Integral

IKASL generalization is used to generally represent the learning outcomes produced at each learning phase. Generalization phase of the IKASL model aggregates learning outcomes represented by the winner node and its neighborhood nodes. This novel adaptation serves both key features of continuous learning and knowledge acquisition. For continuous learning it is important to extract maximum knowledge from the dynamic feature map as these outcomes form the basis for subsequent learning.

The contribution to the weight of the generalized node from the winning node should be maximal and it should be exponentially reduced when going outwards from the winning node. To calculate the weights of generalized nodes, fuzzy integral [9] is used. The fuzzy integral is based on the concept of a fuzzy measure. A fuzzy measure over a set X is a function of the form,

$$g : 2^x \rightarrow [0,1]$$

A fuzzy measure assigns a real value between 0 and 1 for each subset of X

Sugeno Fuzzy Integral

Let g be a fuzzy measure on X and h : X→ [0,1] a function. The Sugeno integral of h with respect to g is,

$$\int h \circ g = \bigvee_{i=1}^{n} \left(h(X_{\Pi_i}) \bigwedge g(A_i) \right) \quad (4)$$

where, $h \rightarrow h(X_{\Pi_1}) \geq h(X_{\Pi_2}) \ldots\ldots \geq h(X_{\Pi_n})$ where $h(X_{\Pi_1})$ is the most important and will be the least important value. Minimum and maximum operators are denoted by the symbols ⋀ and ⋁, respectively.

$$A_i = \{X_{\Pi_1}, X_{\Pi_2}, \ldots\ldots, X_{\Pi_n}\} \quad i = 1,2, \ldots\ldots\ldots, n \quad (5)$$

Application of Fuzzy Integral in IKASL generalization phase,

1) Neighborhood is defined from the winner node and Euclidean distance for each node from the winner is calculated.

2) Sugeno λ measure is defined to be a small constant value

3) Calculate the Sugeno Fuzzy Integral Weight of the generalized node using the arrays of normalized [0-1] weight values and the (1-distance) values.

In contrast to taking average of the winner node and the neighboring nodes, fuzzy integral provides a more intuitive representation of the knowledge encoded in the winner node and the neighboring nodes. Fuzzy integral gradually decreases the contribution of the individual nodes to the final result as the node distance between the winner node and respective node increases. The effectiveness of fuzzy integral can be seen in Fig. 4, as fuzzy generalization has led to a better clustering in contrast to average generalization.

Though fuzzy integral has been in the initial version of the IKASL, it had been published before especially focusing on textual data. In this paper the fuzzy integral based generalization scheme is implemented with other modifications to stabilize and improve the IKASL and value demonstrated with several datasets.

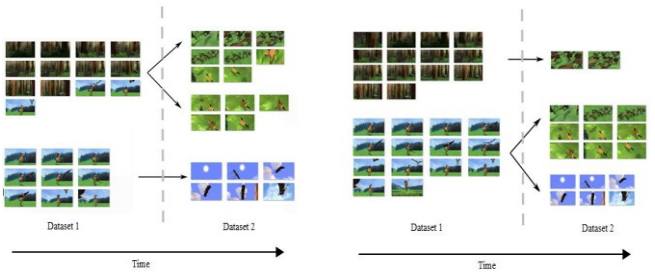

Fig. 4. Left hand side shows the clusters obtained by using 'average' as a generalizing technique. Right hand side shows the clusters obtained by using fuzzy generalization

E. Extended IKASL Algorithm

Notation used in the algorithm are, 'n' - number of data sequences to learn, 'D' - number of dimensions in the data sequence, 'Li' - i-th learning phase, 'Gi' - i-th generalization phase, 'HT' – Hit threshold, 'α' - learning rate for weight adaptations, 'SF' - spread factor controls growth of the GSOM, 'GT' - growth threshold, decides the spread of the network based on D and SF, TEi - total quantization error of node i, 'DTi'- Disparity threshold for Gi

Initialization

1) Initialize weight vectors of the four starting nodes with random values

2) Determine neighborhood size for weight adaptations, neighborhood size for hit node aggregation, learning rate and growth threshold

3) Apply spread factor transformation and calculate new spread factor for 1st layer

Learning

Self-Organization

4) A random input vector is selected and fed to the network of nodes.

5) The node closest to the input vector is identified using Euclidean geometry, find node q′ such that

$$\|v - w_{q'}\| \leq \|v - w_q\|$$

where v, w are the input and weight vectors, respectively, q is the position vector for nodes, and N is the set of natural numbers.

6) The network adapts to the input vector by modifying weights in q′ and nodes in the defined neighborhood.

$$w_j(k+1) = \begin{cases} w_j(k); \ j \in N_{k+1} \\ w_j(k) + LR(k)\left(x_k - w_j(k)\right); \ otherwise \end{cases} \quad (6)$$

where, the learning rate LR(k), k ϵ N is a sequence of positive parameters converging to zero as k →∞. $w_j(k)$, $w_j(k+1)$, are the weight vectors of the node before and after the adaptation, and N_{k+1} is the neighborhood of the winning neuron at $(k+1)^{th}$ iteration.

7) Increase the error value of winner (the difference between the input vector and the weight vectors)

Node Growth

When TEi > GT, if I is a boundary node,

8) Grow new nodes and initialize weights to match neighborhood node weights

If not,

9) Distribute weights to neighbors

Smoothing

10) Reduce learning rate and weight update neighborhood size

11) Resend inputs and adapt weights just as in learning phase

Generalization

12) Identify nodes with hit values above HT, with respective neighborhood nodes

13) Aggregation will be performed using Fuzzy integral, as explained in section III.

Learning from a Generalization layer

The generalization layer preceding L1 will form the basis for all learning phases after the L1 phase. An additional step is required to determine which aggregate node will generate the network topology for each input vector in the new data sequence. Moreover in order to determine whether an input is dispersed significantly from the aggregate node, disparity threshold (DT) is used.

14) All input vectors are fed sequentially, to the nodes of the generalization layer. The aggregate node closest to the input vector is identified using Euclidean geometry,

15) Winning node q′ is assigned the input vector. For all learning epochs to follow, node q′ will develop a feature map representing the set of assigned input vectors.

Each aggregate node will produce a feature map with a single node containing a weight vector identical to that of the aggregate node and map growth will start from this single node. Here the learning rate modification will be applied to prevent learning rate from being negative.

IV. Experiment Results

Several experiments were conducted to validate the necessity and value of incremental learning that can be achieved by the IKASL algorithm. These experiments were focused on answering questions such as,

- Can the algorithm learn incrementally without overwriting past knowledge?
- How does the algorithm behave when it encounters a new knowledge which it has not encountered earlier?
- Can the algorithm incorporate existing knowledge to acquire new knowledge more accurately?

In previous work, several experiments have been conducted to validate functionality of the original IKASL algorithm using images extracted from a cartoon sequence [10]. However, this study presents results of more thorough and comprehensive experiments conducted using real-world images. In contrast to cartoon images, real world images contain lot more details (e.g. variations in color and texture) and unnecessary information (i.e. noise, blur). Such characteristics of real-world images make it difficult to learn/cluster images in an unsupervised manner. Assessing IKASL algorithm with real-world images is very important, as the ultimate goal is to analyze real-world data (e.g. medical, surveillance).

A. Experiment 1

In the first experiment, dataset comprises 3 image sets extracted from several video footages [11], [12], [13]. There are 4 types of entities; a human face, a forest, a beach and an urban scene. Image set sequences are designed in such a way that each type of images evolves as learning period increase, as depicted in Fig. 5.

Fig. 5. Sample sets of the image dataset used. The images evolve as the sequence number increases. For example, observe how the child has grown as the sequence number increases.

Fig. 6 depicts an experimental result obtained by employing previously described dataset using IKASL algorithm. The experiment consists of 3 learning periods. In each learning period, an input image sequence is fed to the algorithm. PHOG descriptor [14] was used for feature extraction. PHOG descriptor is well-suited for this experiment as it can capture both local shapes and spatial layout [14].

In the 1st learning period (Learn cycle 0) images are clustered to 2 major clusters C1 and C2. C1 cluster has a mixed knowledge of a baby's face and trees in winter. C2 cluster contains images of an urban scene. Therefore, it can be argued that C1 doesn't have the required knowledge to distinguish between humans and trees yet.

In the 2nd learning period several interesting events take place. It can be observed that, C1 cluster has been split into 2 clusters where one cluster (C3) has acquired knowledge of the face and another cluster (C4) has acquired knowledge about trees. This particular observation proves that, IKASL algorithm is capable of employing previous knowledge to shape new knowledge to be more accurate. Cluster C3 represents the same person as C1 but grown few years than before. Furthermore, the knowledge about the trees in the winter season has produced a new cluster which contains trees without snow. Thus, it can be seen that, IKASL algorithm is able to capture incremental changes in the data.

In the 3rd learning period, growth of C5 cluster from C2 can be observed. C2 cluster represent an urban scene in the daylight and C5 represent the same scene at night. It can be seen that though any urban scenes hasn't been provided in the 2nd learning period, previous knowledge is retained.

B. Experiment 2

Image collection in Fig. 7 consists of natural sceneries which has different dominant colors. Input is fed to the IKASL algorithm as 4 input sequences. The experiment was based on the dominant colors present in the images.

The feature extraction is done as follows. The HSL color space is divided in to 27 bins. Black (light level less than

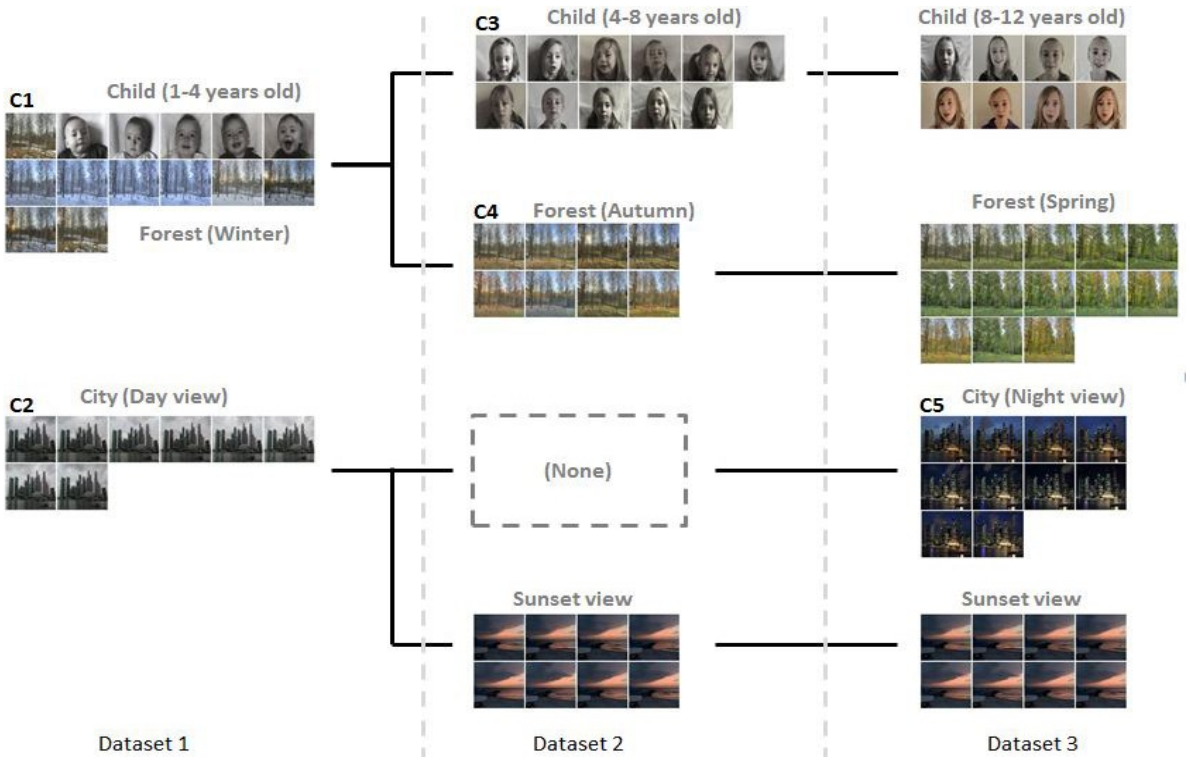

Fig. 6. Experiment results for the image dataset comprising a human face, a forest, a beach and an urban scene. PHOG Descriptor was used to generate feature vectors with following parameters; number of bins = 6 and level = 1. Following parameters were used for the IKASL algorithm. Spread Factor: 0.5, Neighborhood Radius: 2, Learning Rate: 0.3, Iterations: 100

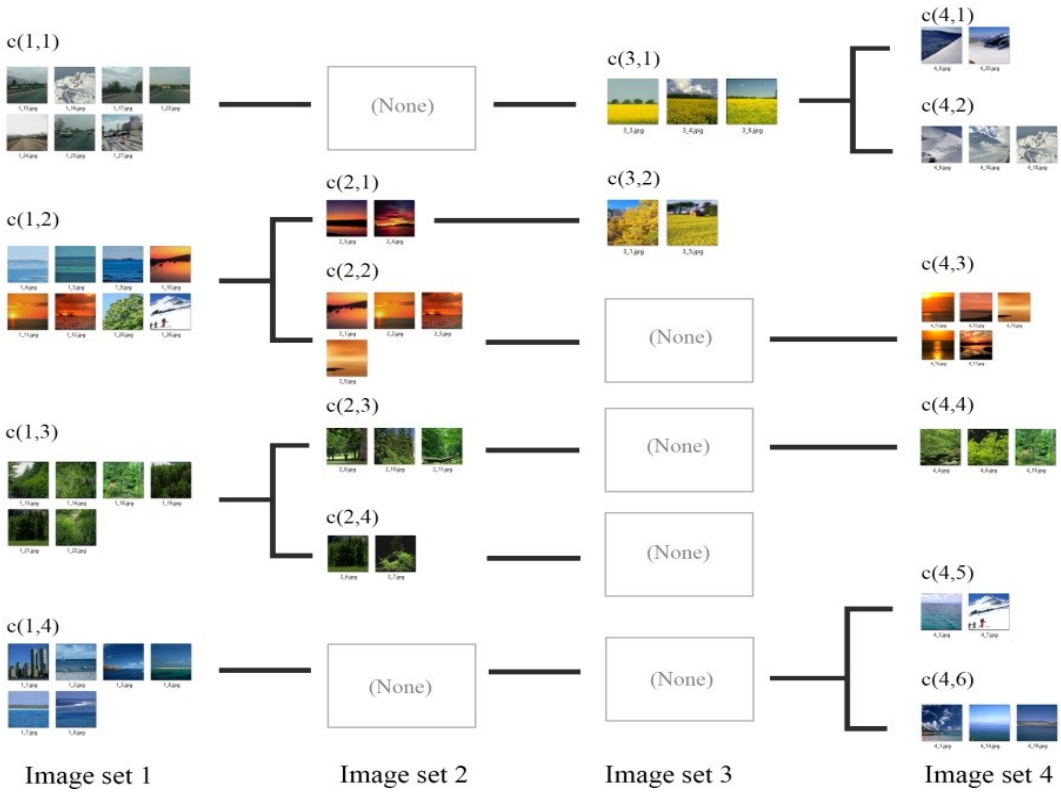

Fig. 7. IKASL output for a natural scene classification. Set 1 – grey, blue, green, red; Set 2 – red, orange green, - no blue, no grey; Set 3 – blue, grey, yellow, grey; Set 4 – grey, blue, green, orange. Existence of colors is used for feature extraction of images. Following parameters were used. Spread Factor: 0.5, Neighborhood Radius: 2, Learning Rate: 0.3, Iterations: 100

0.2), white (light threshold more than 0.9), gray (less than 0.25 saturation threshold) and other colors are split into 24 bins using the hue value (15° for each bin (out of 360°)). Then select the top 8 most dominant colors exist in the image.

Image set 1 consists of images in Grey, Blue Green and Red. As depicted in Fig. 6, result displays a clear classification of different colors into separate clusters. But it can be seen that C(1,2) has mixed knowledge, evident by different types of images in that cluster.

In the next learning period, C(2,1) and C(2,2) which has grown from C(1,2) contains orange/red dominated images. Also, C(2,3) and C(2,4) (green-dominant) clusters has grown from previous green-dominant cluster C(1,3). It can be clearly seen that 2nd learning period has a more refined knowledge about images compared to the 1st learning period. Furthermore, it can be observed that, after the 1st learning period, blue color appears only in the 4th learning period. Yet, IKASL algorithm has recognized that it has acquired knowledge about blue color in the 1st learning period. This particular fact proves that, the algorithm is capable of retaining knowledge without any disruption from new knowledge

Similarly, algorithm has continued to learn about new data based on the past knowledge it possesses.

V. Discussion And Future Work

In this paper, we have discussed the importance of incremental learning by employing a novel unsupervised incremental learning algorithm known as IKASL. IKASL algorithm is both stable and plastic and is capable of performing complex learning functions which can be observed in humans. IKASL algorithm is an n-layer structure where each layer comes into existence as new learning periods occur. Function of IKASL algorithm comprises two main phases; learning phase and generalization phase.

The IKASL algorithm can be further enhanced by employing several extensions. Introduction of the Disparity threshold helps to identify outliers in the input set and prevent affecting existing knowledge of nodes. Increasing spread factor in the initial learning phase ensure that there is a significant spread in the feature map(s) regardless of the spread factor specified by user. Learning rate modification ensures a positive learning rate throughout the learning process. Finally, fuzzy integral reflects the primary and secondary outcomes better than obtaining simple average of nodes.

Moreover, several experiments were conducted by running the IKASL algorithm for several datasets. It is evident from the experimental results that the algorithm is capable of capturing incremental changes in data and learning without forgetting past information. In addition, IKASL algorithm is capable of using previous knowledge to refine newly acquired knowledge. Thereby, we have shown the real world value and necessity of incremental learning.

Though IKASL algorithm is capable of powerful and complex learning functions, there are several extensions that could transform the IKASL algorithm to be capable, more powerful and more complex learning functions.

First, IKASL algorithm is capable of retaining past knowledge unaffected by subsequent knowledge acquisition. However, IKASL algorithm does not differentiate between past knowledge recently activated by new knowledge and knowledge that has been inactive for a long time. Therefore, by introducing a mechanism to activate and deactivate nodes depending on their activation frequency, more information can be captured during the learning process.

Next, IKASL algorithm currently does not possess the ability to merge nodes with similar knowledge to reduce redundant information that could be accumulated. Therefore, by introducing a merging mechanism to merge redundant nodes in the knowledgebase, algorithm will be able to execute the learning process more efficiently.

Finally, IKASL algorithm is an unsupervised neural network based algorithms. However, learning algorithms tend to perform better when there is some amount of human intervention during the learning process. Therefore by transforming the IKASL algorithm to a semi-supervised algorithm, it will be able to yield more accurate results, as human can intervene with the learning process if the algorithm is misguided due to faulty or noisy data.

References

[1] Giraud-Carrier, Christophe. "A note on the utility of incremental learning." AI Communications 13, no. 4 (2000): 215-223.

[2] De Silva, Daswin, and Damminda Alahakoon. "Incremental knowledge acquisi-tion and self learning from text." In Neural Networks (IJCNN), The 2010 Internation-al Joint Conference on, pp. 1-8. IEEE, 2010.

[3] Hamker, Fred H. "Life-long learning cell structures—continuously learning without catastrophic interference." Neural Networks 14, no. 4 (2001): 551-573.

[4] S. Furao, and O. Hasegawa, "An incremental network for on-line unsupervised classification and topology learning", Neural Networks, vol. 19, pp. 90–106, 2006

[5] H.-P. Kriegel, P. Kroger, and I. Gotlibovich. Incremental OPTICS: Efficient computation of updates in a hierarchical cluster ordering. In Data Warehousing and Knowledge Discovery, volume 2737 of Lecture Notes in Computer Science, pages 224{233. Springer, 2003.

[6] D. K. Tasoulis, G. Ross, and N. M. Adams. "Visualising the cluster structure of data streams". In Advances in Intelligent Data Analysis VII, Lecture Notes in Com-puter Science, pp. 81-92. Springer, 2007.

[7] Kohonen, Teuvo. "The self-organizing map." Proceedings of the IEEE 78, no. 9 (1990): 1464-1480.

[8] Alahakoon, Damminda, Saman K. Halgamuge, and Bala Srinivasan. "Dynamic self-organizing maps with controlled growth for knowledge discovery." Neural Net-works, IEEE Transactions on 11, no. 3 (2000): 601-614.

[9] R.R. Yager, "Element selection from a fuzzy subset using the fuzzyintegral", IEEE Transactions on Systems, Man and Cybernetics, vol.23-2, pp. 467-477, 1993

[10] D. De Silva, "A cognitive approach to autonomous incremental learning," Ph.D. dissertation, Clayton School of Inf. Technol., Monash Univ., Melbourne, Australia, 2010.

[11] "Birth to 12 years in 2 min. 45 sec. Time Lapse Lotte. (The Original)," YouTube video, 2:45, posted by "Hofmeester," April 16, 2012, http://www.youtube.com/watch?v=RtyqS68ViWk.

[12] "Time Lapse of Singapore - Raffles Place - From Day to Night (Again) (Full HD version)," YouTube video, 0:18, posted by "merlion444," Nov 19, 2009, http://www.youtube.com/watch?v=n8-WaKkGrkE.

[13] “One year in 40 seconds,” YouTube video, 0:46, posted by "Eirik Solheim," Dec 26, 2008, http://www.youtube.com/watch?v=lmIFXIXQQ_E.

[14] Bosch, Anna, Andrew Zisserman, and Xavier Munoz. "Representing shape with a spatial pyramid kernel." In Proceedings of the 6th ACM international conference on Image and video retrieval, pp. 401-408. ACM, 2007.