Features Combined From Hundreds of Midlayers: Hierarchical Networks With Subnetwork Nodes.

Chandra Swarathesh Addanki
Dept of Computer Science
Lakehead University
Thunder Bay,Canada
caddanki@lakeheadu.ca

Abhishek Guntaka
Dept of Computer Science
Lakehead University
Thunder Bay, Canada
aguntaka@lakeheadu.ca

Rahul Rachakonda
Dept of Computer science
Lakehead University
Thunder Bay, Canada
rrachako@lakeheadu.ca

Krisha Devi Kocherla
Dept of Computer Science
Lakehead University
Thunder Bay, Canada
kkocherl@lakeheadu.ca

***Abstract*—In this project, we utilize an Online Sequential Learning algorithm for single feed forward network with features combined from hundreds of mid layers, the algorithm is called hierarchical online sequential learning algorithm (H-OS-ELM) and can learn data block by block (chunk by chunk) with fixed or varying block size, we believe that the heterogeneous selectivity of neurons in top layers which consists of encoded distributed information produced by the other neurons offers better computational advantage over inference accuracy. Thus this paper proposes a hierarchical model framework combined with Online-Sequential learning algorithm, Firstly the model consists of subspace feature extractor which is basically a sub-network neuron, which is formed by combination of other neurons, using these features from the feature extractor in first layer of the hierarchical network we get rid of irrelevant factors which are of no use for the learning and iterate this process to recast the the sub-features into the hierarchical model to be processed into more acceptable cognition. Secondly by using OS-Elm we are using non-iterative style for learning we are implementing a wider and shallow network which plays a significant role in generalizing the overall performance which is directly in turn boosts up the learning speed.**

***Index Terms*—OS-ElM,Sub-Space Feature extractor,Iterative learning, chuck-by-chuck learning.Image Recognition**

I. Introduction

With the rapid development and advancement in the field of deep learning we have seen the rise of various models such as FeedForward Neural Networks. Convolutional Neural Networks, Recurrent Neural Networks (RNN's), Conditional Random Fields (CRF's) . In fact the development of Deep Learning has started way back in 1980's , In the recent times we have seen various Deep Learning models such as Alex Net,Amoeba Net , RNN (Recurrent neural network) ,these ,models have advanced object recognition by discovering the hidden structures in high dimensional data, the only restrictions with these models is that they are iterative model of learning and due to this particular reason the training time is quite slower , further more because of the Back propaganda used in this kind of iterative learning the model suffers from the false local Minima and vanishing gradient problems and the change in learning rate has direct consequences with the accuracy, there are various kinds of Non-iterative learning models which can solve these problems such as SVM(Support Vector Machines) , Random Forest But these learning strategies are rarely explored .

Furthermore with the development of Deep Learning models more layers are added to the hierarchy, but with the increase in the layers the computational cost is also increased and it will take long time to train the model , making it very harder during inference stage so it motivates us to design an hierarchical model which is computational efficient and can provide us with performance that is not much less than the model which learns features from end to end training. Deep learning is being improved in an exponential rate in the past few years and there are various methods that have reached their Maximum potential and the majority of them use iterative method of learning and this method has been a paradigm for training a hierarchical neural networks, the learning effectiveness and learning speeds of this method of learning is has not yet reached the required amount. Non-iterative learning on the other hand has not been explored to a big extent.

II. Description

A. Feature extraction from hundreds of mid-layers

The number of layers of the network that we relate also has a huge impact on the execution speed and the accuracy of the algorithm (complexity of the algorithm),In this project we are implementing the usage of deep three-general layer learning framework [1] to begin with.This is a modification over the existing single layer feed forward network, this single layer feed forward network has a general hidden layer in which there exists a sub-network of neurons, this sub-network plays a crucial role in feature generation. Hierarchical Method proposed below is the base on which we will be extending it to sequential learning. First we will be discussing the layers and operations in hierarchical method.

B. Online Sequential Learning algorithm :

This algorithm is used for single hidden layer feed forward networks with additive or radial bias function (RBF), It can learn data one by one or chunk by chunk (varying or fixed chunk size, This is mainly used for batch learning.Activation functions used in this algorithm are any bounded non-constant piece wise continues functions (for Additive nodes) and internal piece wise continuous functions (for RBF nodes). In this method we can only choose no.o.nodes, no other parameter is chosen manually. Detailed explanation of OSELM is following

OSELM (First phase):

- Initial training phase In this step a chunk of training data N is necessary to learn an initial model, identical to batch ELM, the input weights and bias of the basic model are randomly generated and the initial output weight is calculated based on the least square solutions.

$$\beta_0 = P_0 H_0^T T_0, P_0 = (H_0^T H_0)^{-1} \tag{1}$$

OSELM (Second phase):

- Sequential learning phase As the new set of training data arrived, the (k+1)th chunk of data $n_{k+1} = \{x_i,\ t_i\}\ where\ \ k\ \geq 0\ and\ N_{k+1}$ denotes the size of (k+1)th chunk, the partial hidden layer output matrix H_{k+1}, is calculated firstly. Then the output weight matrix β_{k+1} with T_{k+1} and β_k is computed as follows $\beta_{k+1} = \beta_k\ + P_{k+1}\ H_{k+1}^T(T_{k+1}\ - H_{k+1}\beta_k\ )$ where, (2) is the Moore- Penrose Pseudoinverse

$$P_{k+1} = P_k\ - P_k H_{k+1}^T(\mathrm{I} + H_{k+1} P_k\ H_{k+1}^T)^{-1} H_{k+1} P_k \tag{2}$$

III. Proposed Methods:

A. Motivation for proposed methods: Subspace feature removal from channels

Hierarchical Method proposed in Fig 1 is the base on which we will be extending it to sequential learning. First we will be discussing the layers and operations in Hierarchical method.

The structure of single-layer network is shown in Fig 1 figure. Mathematically, the output of the single-layer network (Fig. 1) f with d neurons can be represented accordingly.

$$f = g(x, a, b), a\ =\ [a1, ...,\ ad\ ] \epsilon R nd, b \epsilon R \tag{3}$$

Here in Fig 2 we have M samples having n dimensions, x is the input data and a, b are input weights and biases of hidden layer and g(.) represents activation function and a1 is input weight of specific neuron.

where x ϵ R M×n represent the input data which includes M samples with n dimensions, a, b represent input weights and biases of the hidden layer, a1 ϵ Rn represents the input weight of a neuron, and g(•) denotes the activation function of the hidden layers.

Furthermore, for multi-layer network structure, the output of each hidden layer can be shown in Fig 2 as follows. Next is the

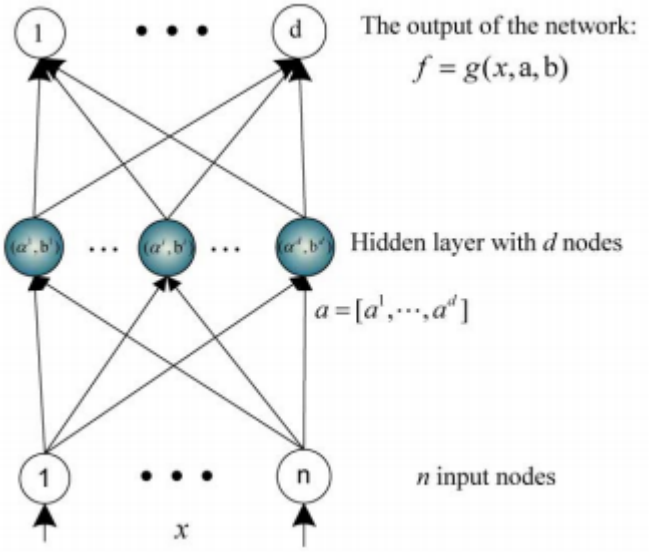

Fig. 1. Structure of single layer feed forward network.The general structure of a single layer feed forward net-work with input nodes,a hidden layer of nodes and the final layer.

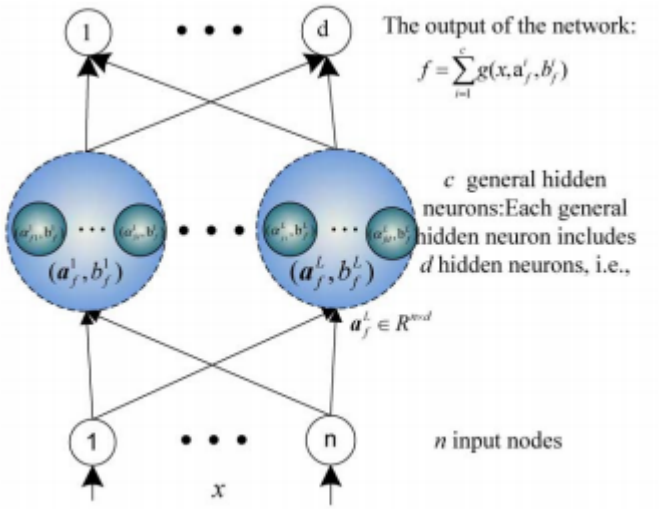

Fig. 2. Single layer feed forward network with sub-network nodes. This is the single layer feed forward network that consists of a hidden layer which consists of a sub-network nodes 2

multi layer network structure, the output of this architecture of each hidden layer can be shown in Fig 3

$$H_f^i = g\left(H_f^{i-1}.\ \left[a^1, \ldots, a^d\right]^T + b\right) \tag{4}$$

In equation 4 H_f^i is the output of the i^{th} layer and the H_f^{i-1} is the output of the $(i-1)^{th}$ layer. Now we will take the idea of this architecture and create a new multi layer network architecture with subnetwork nodes (see Fig. 3). So, now the output of this single-layer network with c subnetwork nodes

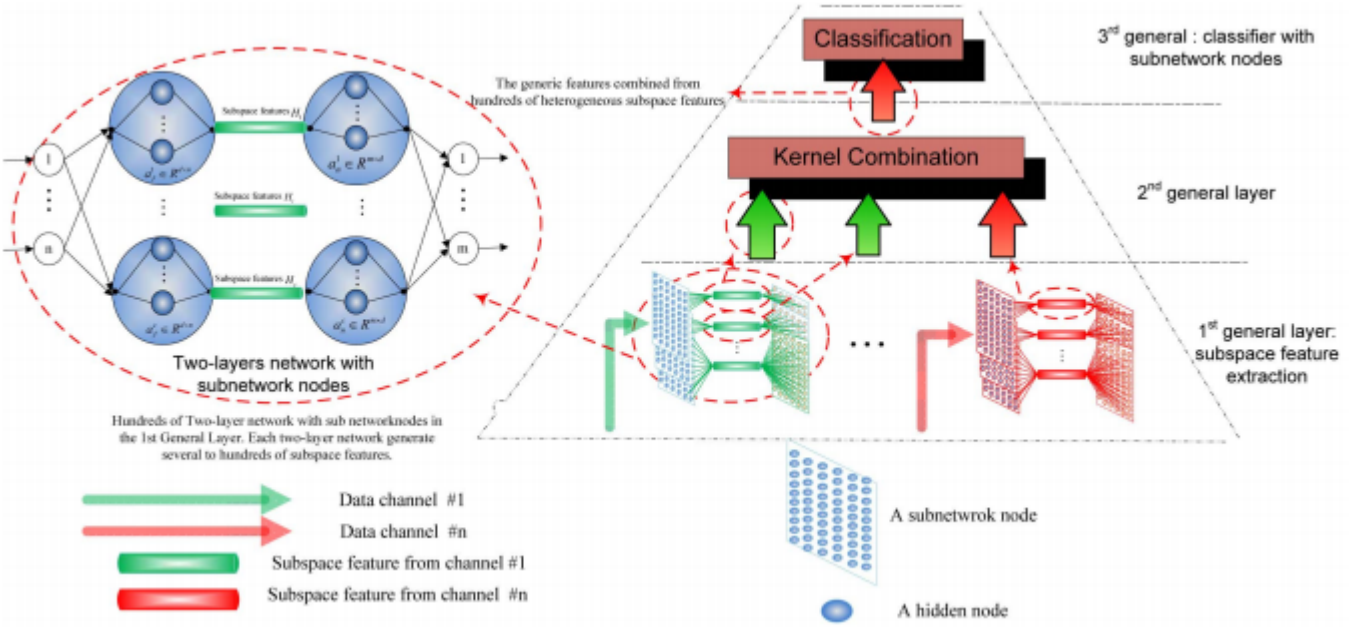

Fig. 3. Method proposed in the paper for the multiple data channel structure.

can be represented as in eq 5

$$f^c = \sum_{i-1}^{c} g(x, a_f^i, b_f^i),\ a^i \ \epsilon\ R^{d \times n} \tag{5}$$

Structural differences between a single layer feed forward neural network(SLFN) and SLFN with subnetwork neurons

- Neuron itself is constituted from other neurons which is called a subnetwork node.
- Subnetwork node is designed as a feature extractor without any activation function inside.
- Subnetwork nodes are not fully connected unlike network connection.
- Features are generated from hundreds of subnetwork nodes are combined in a heterogeneous manner for optimal feature extraction.
- Heterogeneous features are obtained by combining subspace features of different classes.
- Proposed architecture has a wider and more shallower architecture which helps getting good performance with faster learning speeds.

B. Motivation for OSELM:

Traditional ELM which was used in the paper: Calculated subnetwork hidden nodes by pulling back the system residual error to hidden layers. What's more, it ought to be seen that not at all like different ELMs, the input weight is a matrix .there are a few intriguing focuses which should be referenced. Standard ELM strategy is a subnetwork hidden node itself can be framed by a few several hidden nodes Unlike other ELM strategies, these subnetwork hidden nodes are determined and not produced arbitrarily. In this way, both subnetwork hidden nodes and output weights lead not exclusively to the smallest training error, yet in addition to the littlest standard. Moreover, the quantity of hidden nodes(L) and dimension of outputs (m) are independent, however the quantity of concealed hubs in each subnetwork neuron should rise to the component of yields.It likewise infers that each subnetwork hidden node can be considered as a standard Single hidden layer feedforward neural network.

Proposed OS-ELM method: OS-ELM begins from the batch learning extreme learning machine (ELM) created for SLFNs with additive substance and RBF hubs. The exhibition of ELM has been assessed on various benchmark issues from the function regression and classification regions. Results show that contrasted and other gradient-descent- based learning algorithm (counting BP calculations) ELM gives better speculation execution at higher learning speed and the learning stage in numerous applications is finished in no time. OS-ELM with RBF hubs, the focuses and widths of the nodes are randomly generated and fixed and afterward, considering this, the yield output weights are analytically decided. Dissimilar to other consecutive learning algorithms which have many control parameters to be tuned, OS-ELM just requires the quantity of hidden nodes to be indicated.

C. First General Layer: Subspace Feature Extraction

Step 1: Here we have M training samples $x_k,\ y_k\ _{k=1}^{M}\ , x_k \in R^n$ obtained from a continuous system and we will set subnetwork node index equals to 0(c=0).

Step 2 : Now we will generate a subnetwork node randomly as:

$$c = c + 1, H_f^c = a_f^c.x + b_f^c \tag{6}$$

Here H_f^c is the C^{th} subspace features and $a_f^c \in R^{d*n}$ and $b_f^c \in R$ is a random subnetwork neuron.

Step 3 : For getting the desired targets Y, we will calculate the subnetwork in the second layer (a_f^c, b_f^c) by following way:

$$a_h^c = y.\left(H^{cT}{}_f\left(\frac{C}{I} + H_f^c H^{cT}{}_f\right)^{-1}\right), a_h \quad \in R^{d\ \times m} \tag{7}$$

$$b_h^c = \sqrt{mse\left(a_h.\ H_f^c - \ (y)\right)}, b_n \in R \tag{8}$$

Here b_h^c is bias for the subnetwork node and C is a positive value

Step 4 : Here we will calculate the residual error e as:

$$e = y - (a_h^c.H_f^c + b_h^c). \tag{9}$$

Step 5: Here we will calculate the error feedback data as:

$$P = e.\left((a_h^c)_f^T\left(\frac{C}{I} + a_h^c\,(a_h^c)_f^T\right)^{-1}\right). \tag{10}$$

Step 6: Now update the normalized error feedback data as:

$$P = u(P + H_f^c) \tag{11}$$

where u is a normalized function u(.):→[-1,1].

Step 7: Now we will update the parameters of subnetwork node a_f^c, b_f^c in first layer as:

$$a_{temp} = P.\ \left(x^T\left(\frac{C}{I} + \mathrm{x}\mathrm{x}^T\right)^{-1}\right) \tag{12}$$

$$a_f^c = \ a_f^c + \lambda.(a_{temp} - \ a_f^c\ ) \tag{13}$$

$$b_f^c = \sqrt{mse\left(a_f^c.\ \ x\ \ - P\ \right)}, \quad b_f^c \in R \tag{14}$$

Step 8 : Now calculate the c^{th} subspace features as:

$$H_f^c = a_f^c.x + b_f^c \tag{15}$$

Step 9 : By using the above steps 1-8 generate L-1 subnetwork nodes and obtain the L subspace features.

$$\left\{H_f^1, \ldots..., H_f^C \ldots, H_f^L\right\} \tag{16}$$

D. Second General Layer: Subspace Feature Extraction

Here we will take all the features extracted from the first layer and combine them. To combine the features we will use combination operators. So, the combined feature will be as in the eq 17

$$H^{I\oplus J} = K(H_f^i, H_f^j, \gamma) \tag{17}$$

where $K(H_f^i, H_f^j, \gamma),\ i\ \neq j.$

Here $K(H_f^i, H_f^j,),\ \ i\ \neq j$ is a combination operator.

Given the obtained several subspace features $H_f^1, \ldots\ldots, H_f^c$, the expression is

$$H^{1\oplus 2} = K\left(\ H_f^i, H_f^j, \gamma\right) \tag{18}$$

$$H^{1\oplus 2\oplus 3} = K\left(K\left(H_f^i, H_f^j, \gamma\right),\ H_f^3,\ r\right) \tag{19}$$

$$\vdots$$

$$H^{1\oplus 2\oplus..\oplus c} = \left(\ldots K\left(K\left(H_f^i, H_f^j, \gamma\right),\ H_f^3,\ r\right)\ldots\right). \tag{20}$$

Following are different types of combination operators:

- Plus operator

$$K\left(\ H_f^i, H_f^j, \gamma\right) = H_f^i + H_f^j. \tag{21}$$

- Linear Kernel

$$K\left(\ H_f^i, H_f^j, \gamma\right) = H_f^i\ .\left(H_f^j\right)^T \tag{22}$$

- Radial Bias Function Kernel

$$K\left(\ H_f^i, H_f^j, \gamma\right) = exp(-\gamma\ H_f^i - H_f^{j2}) \tag{23}$$

- Polynomial Kernel

$$K\left(\ H_f^i, H_f^j, \gamma\right) = \left(H_f^i\ .\left(H_f^j\right)^T\right)^\gamma. \tag{24}$$

- Concatenation Operator

$$K\ \left(H_f^i, H_f^j\right) = [H_f^i, H_f^j] \tag{25}$$

E. Third General Layer: Subspace Feature Extraction

The classifier for the final layer is taken from the previous work[13]. So now we have M distinct features combined from the combination operator (H, t) and a normalized function $u(x) : R \rightarrow (0, 1]$, a sigmoid or sine function g, and then for any continuous outputs t, we have the

$\lim_{n\rightarrow\infty} t \ - \ \left(\left(\mathrm{g}\ (\ a_p^1,\ b_p^1,\ H\ )\right).\ \ \beta\ {}_p^1\ +\ \cdots\ +\ \left(\mathrm{g}\ (\ a_p^c,\ b_p^c,\ H\ )\right).\ \beta\ {}_p^c\right) = 0$ holds with probability one if $a_p^c = \mathrm{g}^{-1}((e_{c-1})).\ x^T\left(\frac{C}{I} + \mathrm{x}\mathrm{x}^T\right)^{-1}$

Where, $[\bullet]^{-1}$ in eq 26 represents it is the inverse function. The below are the operations performed and detailed steps will be given in the algorithm below.

Here H_f^c is the C^{th} subspace features and $a_f^c \in R^{d*n}$ and $b_f^c \in R$ is a random subnetwork neuron.

$$b_p^c = \text{sum}\left(a_p^c.\ H - h^{-1}((e_{c-1}))\right)/\ N,\ b_p^c \in \mathbf{R} \quad (26)$$

$$g^{-1}(\cdot)\begin{cases} arcsin(\cdot) & if g(\cdot) = sin(\cdot) \\ -log\left(\frac{1}{(\cdot)} - 1\right) & if g(\cdot) = 1/(1+e^{(\cdot)}) \end{cases}$$

$$e_c = t - u_n^{-1} g(H, a_p^c, b_p^c)$$

$$\beta_p^c = \frac{\left\{e_{c-1}, u^{-1}\left(g\left(a_p^c.H + b_p^c\right)\right)\right\}}{\left\|u^{-1}\left(g\left(a_p^c.H + b_p^c\right)\right)\right\|^2} \quad (27)$$

IV. Algorithm:

A. Proposed Flowchart

OS-ELM with the hierarchical network :

- Training Observations are taken sequentially (one by one or chunk by chunk) with varying or fixed chunk length fed to learning algorithm.
- At any time, only the newly arrived single or chunk of observations are seen and learned.
- After a chunk of training observations are discarded after the respective observations are complete.
- Learning algorithm has no prior knowledge as to how many training observations will be presented.

Explanation of flow chart: The First layer is all about extracting the subspace features and then the boosting phase is applied in step 7 and it uses the calculations for learning in the sequential learning phase. After the subspace features are concatenated in the second layer with mathematical equations. It is sent to the final layer for classification.

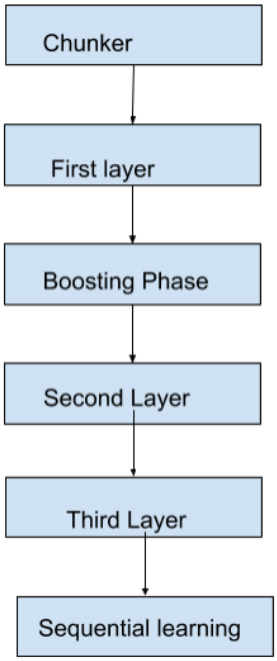

Fig. 4. Proposed Flowchart.

B. Proposed Algorithm

Step-1:

Given N features groups$(Q_1, \ldots\ldots, Q_n)$ generated from the same data set$Q_1 = \left(x_k^1, y_k^1\right)_{k=1}^M, x_k^1 \epsilon R^{n_1}, \ldots., _N = \left(x_k^N, y_k^N\right)_{k=1}^M, x_k^N \epsilon R^{n_N}$,positive coefficient C, and the number of sub network nodes L.

First General Layer: Subspace feature extraction: Step 2: Set n=1 , and let x,y equals Q_1.. While $n < N$ do While $c < L$ do Obtain a subspace feature $H_f^{c+((n-1)\times L}$ bases on steps 1-8. end while Step 3: Boosting Phase: Calculate Moore Penrose pseudo inverse:

$$M_(0) = "(" H_f^(c^T) H_f^{c"})"^(- 1) H_f^(c^T) \quad (26)$$

Calculate output weights matrix

$$\beta_0 = M_(0) H_f^(c^T) \quad (27)$$

return obtained L subspace features $H_f^{c+(1+(n-1))\times L}$.....$H_f^{c+((n-1)\times L}$ from data group Q_n end while return obtain $L * N$ subspace features H_f^1.....$H_f^(LN)$

Second General Layer: Feature combination: Obtained combined features H as

$$H = H^{1 \oplus 2 \oplus (N \times L)} \quad (28)$$

Third General Layer: Pattern Learning

Given concatenated features H, Set c=1, e=t

While $c < L$ do

Step 1: Calculate the cth subnetwotk node (a_p^c, b_p^c), and β_p^c as

$$a_p^c = g^{-1}((e_{c-1})).\ x^T\left(\frac{C}{I} + xx^T\right)^{-1}$$

$$b_p^c = \text{sum}\left(a_p^c.\ H - h^{-1}((e_{c-1}))\right)/\ N,\ b_p^c \in \mathbf{R}$$

$$\beta_p^c = \frac{\left\{e_{c-1}, u^{-1}\left(g\left(a_p^c.H + b_p^c\right)\right)\right\}}{\left\|u^{-1}\left(g\left(a_p^c.H + b_p^c\right)\right)\right\|^2}$$

$$M_{k+1} = M_k - \frac{M_k H_{k+1}^T H_{k+1} M_k}{1 + H_{k+1}^T H_{k+1} M_k}$$

V. Experimental Verification

A. Experiment environment settings:

The image data set, we perform all processing in gray scale even when color images are available. All experiments are repeated a couple of times with different randomly selected training and test images, and the average of per-class recognition rates is recorded for each run Scale-invariant feature transform or the raw images are used for the scene 15. For scene 15 data set we use 100 images per category for training. All the remaining images are used for testing. For the data set scene-15 the feature type is spatial pyramid and there are 4485 images where the training image per category is 100. The category is 15. The number of sub-network neurons is 3. The neurons in each sub-network neuron is 100. The computation time is 20 seconds.

To check the performance of the algorithms in our experiments, tested data sets are chosen from a wide variety of available data sets, including ones with little, medium, enormous, or potentially high dimensions. All databases are preprocessed similarly (utilizing the held-out strategy). Table I shows the training sample test numbers and the class quantities of the relating data sets.

Changes of the entire informational collection are taken with irregular substitution, and preparing pictures (given in Table I) are utilized to make the preparation set. The rest of the informational collection is utilized for the test set. For these picture informational indexes, we perform all processing in gray-scale in any event, when color pictures are accessible.

All analyses are repeated multiple times with various randomly selected training and test images, and the average of per-class recognition rates is recorded for each run. For all image data sets, a few feature descriptor strategies, including CNN highlights, spatial pyramid highlights , various leveled coordinating interest include ,scale-invariant feature change, or raw picture, are utilized for the accompanying informational collections: Caltech101, Oxford Flower 102, Scene15, CIFAR10, For CNNs related highlights, we extricate the features from penultimate layer of the specific network. CNNs have the very similar network architectures for AlexNet, GoogleNet, DenseNet, and VGG. For AlexNet and VGG, we utilized the 4096-dimensional highlights from the completely associated layer of the CNN.

B. Scene recognition

Scene 15 data set contains 4486 gray-value images of which 3860 images are from the 15-category scenes.Following the common experimental settings, we randomly select 100 images per category as training data and use the rest as test data. The Plus operator is used in the combination operator part. In each sub-network node, the number of hidden nodes is set to 100. Our method uses single features with a three-layer network, while some other state-of-the-art approaches combine multiple features, feature fusion methods, and even pre-trained, super-large data sets like Places data set.

C. Caltech 101

The Caltech101 data set contains 9144 images in 102 object categories with the number of images per category varies from 31 to 800. We utilize three subnetwork hubs (100 concealed hubs in each subnetwork hub, i.e., feature dimensionality rises to 100) with five channels Our best outcome is 89.1%, though looked at strategies score 70–80 %. We have tested on caltech 101 with features HMP seperately having 4 subnetwork nodes and combining the spatial pyramid and HMP having 5 subnetwork nodes

D. Cifar 10

The CIFAR 10 data set collection comprises 60000 color images in ten classifications including airplane, bird, automobile, cat, dog, frog, deer, ship, horse, and truck. The number of images per class is 6000. There are 50000 pictures (5000 for each class) for preparing and the rest of the parts for testing.

For the examination This is on the grounds that systems dependent on Deep Learning structure ordinarily give a superior speculation execution, however with extra computational outstanding burdens. For AlexNet and 16-VGG arrange, we utilize a momentum of 0.9 and start with a learning pace of 0.001 with a solitary GTX 1060 GPU. it is anything but difficult to acknowledge that CNN systems, all in all, give a superior presentation while going further, which is steady with the outcomes appeared right now 110-layer ResNet got the best execution among all the CNN models.

In the proposed technique, we utilized four subnetwork nodes(400 hidden nodes in each subnetwork hub) with a solitary channel. Be that as it may, curiously, by utilizing such optimized features , our moderately straightforward model with a few minutes preparing time could additionally help the acknowledgment performance.

E. Structure and Parameters Sensitivity

Our Method With Fixed Structure: Specifications and experimental settings of dataset can be shown in Table I, including number of subnetwork neurons, number of neurons in each subnetwork neuron,and number of data channel. We also sue normal distribtion to randomly produce initial weight

F. Experimental Discussion

In the experiment, we evaluate our proposed strategy on a few generally selected databases including a small data set Scene-15, medium data set Caltech101 and CIFAR10, and a large-scale data . In light of the test results, we sight the accompanying decisions in the result section.

TABLE I
SPECIFICATIONS OF DATASET AND EXPERIMENTAL SETTINGS

Index	*Dataset*	*Feature Type*	*Category*	*Subnetwork neurons*
1	Scene15[a]	Spatial Pyramid	15	3
2	Caltech101[a]	Spatial Pyramid	102	3
3	Caltech101[a]	HMP and Spatial Pyramid	102	5
4	Cifar10[a]	Imagenet Pretrained Deep Feature	10	3

[a]

VI. RESULTS

Index	*Method*	*Dataset*	*Accuracy*	*Time-Taken*
1	Hierarchical Method[a]	Scene15	97.80	46s
2	OS-ELM Combined Method[a]	Scene15	87.9	27s
3	Hierarchical Method[a]	Caltech101	80.01	270s
4	OS-ELM Combined Method[a]	Caltech101	50.20	290s
5	Hierarchical Method[a]	Caltech101(HMP+Spatial)	81.20	290s
6	OS-ELM Combined Method[a]	Caltech101(HMP+Spatial)	49.20	290s

[a]

VII. CONCLUSION

In this paper we present a hierarchical network with subnetwork in it and later combined OS-ELM approach to it evaluated the performance. The problem Is approached in two different ways:

- Heterogeneous features extracted from two layers are combined and sent to the third layer and
- OS-ELM algorithm fused into architecture at boosting phase

there by, attaching to first layer and using the Moore's-Penrose pseudo inverse calculated output in the third layer.Comparing the results of hierarchical networks to other state-of-art methods it gives a better performance. This method can be used as feature extractor(First layer) and a classifier(last layer) and also gives better performance. We have tried to extend the work of the above hierarchical network to see how it performs for sequential learning. So instead of training all samples at once as mentioned in hierarchical architecture we tried to give the sample batch- by-batch for learning. As we can see even though the combined network didn't give the results as better as the original hierarchical network, It gave a decent performance. Further improvements or tweaks in this proposed method can give better performance.

REFERENCES

[1] Yimin Yang, "Features Combined From Hundreds of Midlayers: Hierarchial Networks with Subnetwork nodes", Neural Netw. , Jan. 2019.

[2] Nan Ying Liang, "A Fast and Accurate Online Sequential Learning Algorithm For Feedforward Networks", Neural Networks, Vol 17, No 6, Nov. 2006

[3] J. Schmidhuber, "Deep learning in neural networks: An overview," Neural Netw., vol. 61, pp. 85–117, Jan. 2015.

[4] Y. Yang and Q. M. J. Wu, "Extreme learning machine with subnetwork hidden nodes for regression and classification," IEEE Trans. Cybern., vol. 46, no. 12, pp. 2885–2898, Dec. 2016.

[5] S. Lazebnik, C. Schmid, and J. Ponce, "Beyond bags of features: Spatial pyramid matching for recognizing natural scene categories," in Proc. IEEE Int. Conf. Comput. Vis. Pattern Recognit., New York, NY, USA, Jun. 2006, pp. 2169–2178.

[6] M. D. Zeiler and R. Fergus, "Visualizing and understanding convolutional networks," in Proc. IEEE Eur. Conf. Comput. Vis., Zürich, Switzerland, 2014, pp. 818–833.

[7] K. Simonyan and A. Zisserman. (2014). "Very deep convolutional networks for large-scale image recognition." [Online]. Available: https://arxiv.org/abs/1409.1556v6

[8] Y. Yang and Q. M. J. Wu, "Multilayer extreme learning machine with subnetwork nodes for representation learning," IEEE Trans. Cybern., vol. 46, no. 11, pp. 2570–2583, Nov. 2016.

[9] X. Zhu, "Semi-supervised learning literature survey," Dept. Comput. Sci., Univ. Wisconsin-Madison, Madison, WI, USA, Tech. Rep. 1530, 2006.

[10] G. B. Huang, Q. Y. Zhu, and C. K. Siew, "Extreme learning machine: Theory and applications," Neurocomputing, vol. 70, nos. 1–3, pp. 489–501, Dec. 2006.

[11] X. Z. Wang, Q. Y. Shao, Q. Miao, and J. H. Zhai, "Architecture selection for networks trained with extreme learning machine using localized generalization error model," Neurocomputing, vol. 102, pp. 3–9, Feb. 2013.

[12] J. W. Cao and Z. P. Lin, "Extreme learning machines on high dimensional and large data applications: A survey," Math. Prob. Eng., vol. 2015, pp. 1–12, Mar. 2015.