sustainability

MDPI

A Sustainable Deep Learning Framework for Objects Recognition using Multi-Layers Deep Features Fusion and Selection

Muhammad Rashid[1], Muhammad Attique Khan[2], Majed Alhaisoni[3], Shui-Hua Wang[4*], Syed Rameez Naqvi[5], Amjad Rehman[6], Tanzila Saba[7]

[1,2] Department of Computer Science, HITEC University Taxila, Pakistan; (email: rashidraocomsats@gmail.com, attique@ciitwah.edu.pk)

[3] College of Computer Science and Engineering, University of Ha'il, Ha'il, Saudi Arabia; (e-mail: majed.alhaisoni@gmail.com)

[4] School of Architecture Building and Civil engineering, Loughborough University, Loughborough, LE11 3TU, UK; (email: shuihuawang@ieee.org)

[5] Department of EE, COMSATS University Islamabad, Wah campus, Pakistan (email: rameeznaqvi@ciitwah.edu.pk)

6 Department of Computer and Information Sciences, Prince Sultan University, SA; (email: drrehman70@gmail.com, tsaba@psu.edu.sa)

* Correspondence: shuihuawang@ieee.org, attique@ciitwah.edu.pk;

Abstract: With an overwhelming increase in the demand of autonomous systems, especially in the applications related to intelligent robotics and visual surveillance, come stringent accuracy requirements for complex object recognition. A system that maintains its performance against a change in the object's nature is said to be sustainable; and it has become a major area of research for the computer vision research community in the past few years. In this work, we present a sustainable deep learning architecture, which utilizes multi-layer deep feature fusion and selection, for accurate object classification. The proposed approach comprises three steps: 1) By utilizing two deep learning architectures, VGG19 and Inception V3, it extracts features based on transfer learning, 2) Fusion of all the extracted feature vectors is performed by means of a parallel maximum covariance approach, and 3) The best features are selected using Multi Logistic Regression controlled Entropy-Variances (MRcEV) method. For verification of robust selected features, the Ensemble Learning method named Subspace Discriminant Analysis (ESDA) is utilized as a fitness function. The experimental process is conducted using three publicly available datasets, including Caltech-101, Birds database, and Butterflies database, and the Ten-Fold validation yields the best accuracies of 95.5%, 100%, and 98% for the datasets respectively. Based on the detailed statistical analysis and comparison with the existing methods, the proposed selection method gives significantly more accuracy. Moreover, the computational time of the proposed selection method is better for real-time implementation.

Keywords: Object Classification; Deep Learning; Features Fusion; Features Selection; Recognition

1. Introduction

Object recognition is currently one of the most focused areas of research due to its emerging application in intelligent robotics and visual surveillance [1, 2]. The researchers, however, are still facing problems in this domain for correct object recognition, such as in recognizing an object's shape, and spotting a minor difference among several objects. Therefore, a sustainable system – the one that maintains its performance against a change in the object's nature – is required for correct recognition of complex objects [3]. Object classification is the key to a sustainable visual surveillance system [4]. Besides the latter, object classification finds its application in numerous domains, including intelligent robotics, face and action recognition, video watermarking, pedestrian tracking,

autonomous vehicles, semantic scene analysis, content-based image retrieval, and many more. We believe that there exist many challenges, like complex background, different shape and same color for different objects, continuously moving objects, different angles, and many more, which demand sustainability, since the unsustainable systems did not work well for complex objects classification [5].

Many techniques were introduced in computer vision to overcome the previously discussed challenges related to complex object nature. They all tried to get the optimal methods that would perform the same for many types of problems, but this was a considerable challenge. Although in the past few decades, the conventional approaches, like Hand-Crafted Features (HCF), were used, as time passed, the objects and their backgrounds became more confusing, which obsoleted their use. Handcrafted features included Histogram of Oriented Graph (HOG) [6], geometric features (GLCM) [7], Scale Invariant Feature Transformation (SIFT) [8], Difference of Gaussian (DoG) [9], Speeded Up Robust Features (SURF) [10], and Texture features (HARLICK) [11]. Recent techniques showed that many approaches were introduced by using different feature techniques – a few of them using the fusion process to get a better representation of an object [12]. However, some recent techniques tried to resolve the issues related to similar objects and appearances, a few of which addressed the problems associated with complex background. Unfortunately, these techniques were unable to recognize the growing complexities of objects and images as well.

In face of the aforementioned challenges, the concept of deep learning has been introduced in this context, which has also shown improved performance against reduced computational time. In relation to this, a large number of convolutional neural networks (CNN) pretrained models have been proposed. This includes AlexNet [13], VGG (VGG-16, VGG-19 [14], GoogleNet [14], ResNet (Resnet-50, ResNet-102, and ResNet-152)[15], and Inception[16]; all these models are trained on the ImageNet dataset. Even with these contributions, however, acceptable accuracy is difficult to achieve. This has given rise to the concept of features fusion [15, 16] – a process of combining several feature populations into a single feature space, which has been adopted in various applications ranging from medical imaging to object classification [17 – 19]. The concept of feature fusion does manage to achieve increased classification accuracy, but only at an increased computational cost. In addition, some of the recent works have shown that the fusion process may add irrelevant features that are not important for the classification task [reference]. It has also been concluded that if the irrelevant features were selected and removed from the fused vector, then the computational time could be minimized with an increased accuracy.

Feature selection can be categorized into three: filter based, wrapper based, and embedded. The filter based selection selects the features from subsets independently. The wrapper based methods initially assume the features, and then selects them based on predictive power. The embedded selection initially utilizes the selection in the training phase, which enjoys the advantages of both filter based and wrapper based [17]. Some of the famous feature selection techniques include Principle Component Analysis (PCA) [18], Linear Discriminant Analysis (LDA) [19], Pearson Correlation Coefficient (PCC) [20], Independent Component Analysis (ICA) [20], Entropy Controlled [21], Genetic Algorithm [12] based, and many more.

In this work, an entire sustainable framework based on a deep learning architecture is proposed. While we summarize our challenges, and highlight our contributions in response to those in Section 3, the details on the proposed framework are explicitly given in Section 4. Section 5 presents the simulation results, before we conclude the manuscript in Section 6. In what follows, however, we review some of the existing related works, in Section 2.

2. Related Work

Many strategies are performed for image classification, as investigated in the area of computer vision and machine learning. Object categorization is the most emergent field of computer vision because of its enormous applications, for example, video surveillance, auto assisted vehicle frameworks, pedestrian analysis, automatic target recognition, and so on. In the literature, very few

fusion-based techniques are presented for the classification of complex objects. Features fusion is the process of combining two or more feature spaces into a single matrix. By fusion, there is a chance to get a higher accuracy vector having properties of multiple feature spaces. Roshan et al. [22] presented a new technique for object classification. They applied the presented algorithm on VGG-16 architecture and performed training from scratch. Also, they applied the transfer learning on top layers. They utilized the Caltech101 dataset and achieved an accuracy of 91.66%. Jongbin et al. [23] introduced a new DFT based technique for feature building by discarding the pooling layers among fully connected and convolutional layers. Two modules are implemented in this technique. The first module known as DFT was performing the replacement of max pooling from the architecture by a user-defined size pooling. The second module, known as DFT+, was the fusion of multiple layers to get the best classification accuracy. They achieved 93.2% classification accuracy on the Caltech-101 dataset using the VGG-16 CNN network and 93.6% accuracy on Caltech-101 using the Resnet-50 model. Qun et al. [24] used a pre-trained network with associative memory banks for feature extraction. They extracted the features using ResNet-50 and VGG-16. Later on, K-Means clustering was used on memory banks to perform unsupervised clustering. Qing et al. [25] presented a fused framework for object classification. They extracted CNN features and applied three different types of coding techniques on to fused vector. Two pre-trained models, namely VGG-M and VGG-16, were used for feature extraction. After feature extraction from 5-Conv-Layer, PCA based reduction was applied, and features were fused into a final vector using proposed coding techniques. Results showed that the introduced technique reported an improved accuracy of 92.54% by using their third coding technique on the Caltech-101 database. Xueliang et al. [26] presented a late fusion based technique for object recognition. Three pre-trained networks, namely AlexNet, VGGNet, and ResNet-50, are used. Firstly, they evaluated that the middle-level layers of CNN architecture contain more robust information for visual representation, and then features were extracted from mid-level layers. Feature fusion from these three models showed improved result and reported 92.2% accuracy on Caltech-101 dataset. Hamayun et al. [27] proved that the most robust features are extracted from fully convolutional layer-6 (FC-6) instead of FC-8. In the presented approach, they exploit the CNN output and modify it at a middle-level layer instead of the deepest layer. VGG-16 and VGG-19 pre-trained models are used to illustrate the proposed technique. They extracted 4096 features from the FC-6 layer and then applied reduction using PCA. For the experimental process, they used Caltech101 and attained an accuracy of 91.35% using reduced features from layer FC-6. Mahmood et al. [28] gave an idea to object detection and classification using pre-trained networks (ResNet-50 and ResNet-152). After feature extraction, they performed features reduction using PCA. The Caltech-101 database was selected for evaluation and achieved an accuracy of 92.6%. Emine et al. [29] used Convolutional architecture for fast feature embedding (Caffe) for object recognition. The Caltech-101 dataset was used to test the proposed technique. Results showed that 300 images were tested from the whole dataset. From those, 260 were correctly classified, and 40 were misclassified. Chunjie et al. [30] introduced a new technique to handle the drawbacks occurred by local features named as Contextual Exemplar. The technique followed by three phases. In the first phase, they combined the regions-based image, the second phase was constructing the relationship between those regions, and the third phase was using the connection of those regions for semantic representation. They selected 1000 features and achieved an accuracy of 86.14%. Rashid et al. [31] focused on multiple features fusion and selection of best of them for efficient object classification. They used VGG and Alexnet pre-trained models for CNN feature extraction and SIFT as point features extraction. Both types of features are fused by a simple concatenation approach. Later on, they implemented an entropy-based selection approach and achieved an accuracy of 89.7% for the Caltech101 dataset. **Nazar** et al. [32] fused HOG and Inception V3 CNN features and improved the existing accuracy up to 90.1% for Caltech-101 dataset.

3. Challenges and Contributions

The computer vision research community is still facing various challenges for object classification, and most of them are due to the complex nature of objects. We do realize that it is not

an easy task to efficiently classify objects into their relevant categories. In order to be able to tackle the challenges facing the community and achieve the required accuracies, in this work, we propose a deep learning architecture based framework for object classification with improved accuracy. The highlights of the framework are as follows:

- It uses two pre-trained deep learning architectures, namely- VGG19 and Inception V3, and performs transfer learning (TL) to retrain the selected datasets. The FC7 and Average Pool layers of the CNN are utilized for feature extraction.
- A parallel maximum covariance (PMC) technique is proposed for the fusion of both deep learning feature vectors.
- While the Multi Logistic Regression controlled Entropy-Variances (MRcEV) method is employed for selecting the robust features, the Ensemble Subspace Discriminant (ESD) classifier is employed as a fitness function.
- A detailed statistical analysis of the proposed method is conducted, and compared with recent techniques to examine the stability of the proposed architecture

4. Materials and Methods

The proposed object classification architecture is presented in this section with the detailed mathematical formulation and visible results. As shown in **Figure 1**, the proposed architecture consists of three core steps: deep learning feature extraction using TL, fusion of various model features, and selection of the robust features for final classification. In the classification step, the ESD classifier is used, and the performance is compared with other learning algorithms. The details of each step, depicting in this figure, are discussed below.

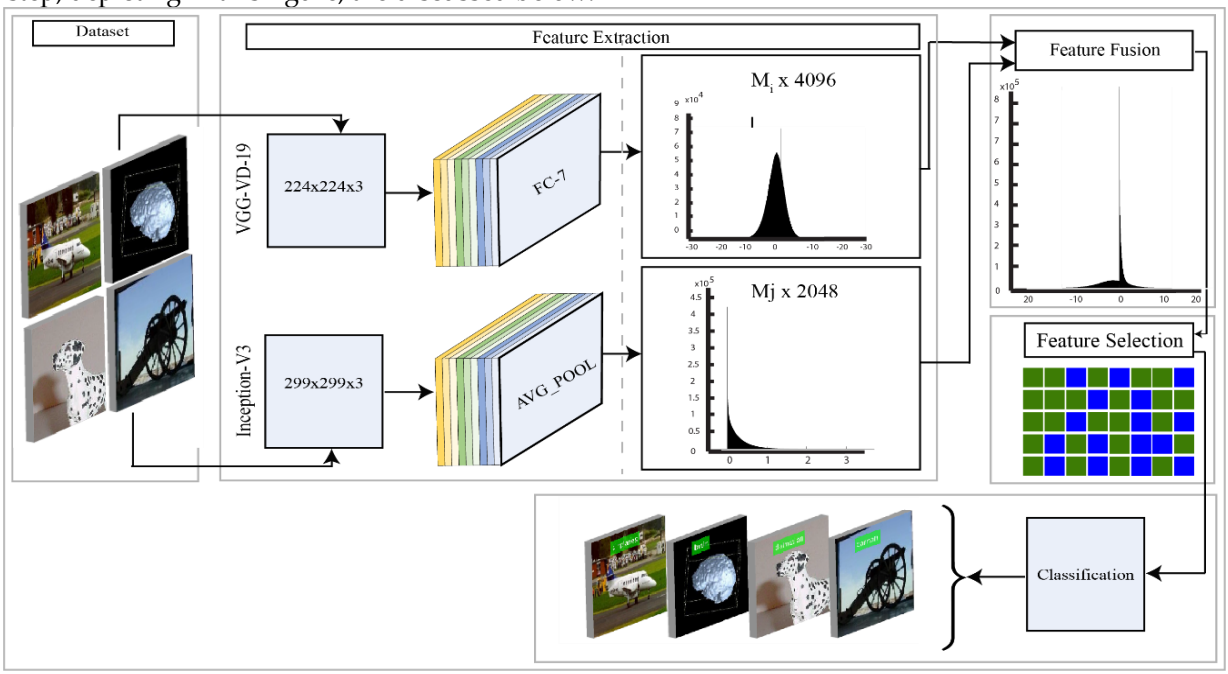

Figure 1: *Proposed deep learning architecture for objects classification*

4.1. Deep Learning Features Extraction

Since the past two decades, the deep learning has proven itself as the best approach for image recognition and classification [7, 8, 33, 34]. CNN is a type of deep learning and involves a series of layers. A simple CNN model consists of convolution and pooling layers. A few other layers are activation named ReLu and feature layer named fully connected (FC). The first layer of a CNN is known as the input layer. This layer takes an image as the input, and convolutional layer computes the neurons' response. The latter is calculated by the dot product of weights and smaller regions. While the ReLu layer helps in the activation function, the pooling layer between convolution layers removes the inactive neurons for the next phase. Finally, the high-level features are computed using FC layers, which are classified through Softmax [8]. In this work, we are using two pre-trained CNN

models, namely- VGG19 and Inception V3, for feature extraction. In what follows, we present a brief description of each model.

VGG19: VGG-19 [35] consists of sixteen convolutional layers, 19 learnable weights layers, which are utilized for transfer learning, three FC layers, and an output layer. This model is already trained on the ImageNet dataset. The input size for this model is $224 \times 224 \times 3$, as given in Table 1. The learnable weights and bias of the first convolution layer is $3 \times 2 \times 3 \times 64$ and $1 \times 1 \times 64$. The total learnable at this layer is 1792. For the second convolution layer, the total learnable is 36928. This layer extracts the local features of an image.

$$V_i^{(M)} = B_i^{(M)} + \sum_{k=1}^{n_1^{(M-1)}} \psi_{i,k}^{(M)} \times h_k^{(M-1)} \quad (1)$$

Where, $V_i^{(M)}$is the output layer L_y, $B_i^{(M)}$ is the base value, $\psi_{i,k}{}^{(M)}$ denotes the filter mapping the kth feature value, and h_k means the $M-1$ output layer. The learnable weights and bias of the first FC layer are 4096×25088 and 4096×1. The dropout layer is added between FC layers, where the dropout rate is 50%. For FC layer 7, the total learnable is 16781312, and learnable weights are 4096×4096. For the last FC layer, the total learnable is 4097000, and learnable weights are 1000×4096. Hence, when the activation is applied, it returns a feature map vector of dimension $1 \times 1 \times 1000$. For fully connected layers one and two, the feature map vector dimension is $1 \times 1 \times 4096$.

Table 1: Detailed description of VGG19 pre-trained CNN model

Sr No.	Name	Type	Activation	learnable		Total Learnables
				Weights	Bias	
1	Input	Image Input	224x224x3			
2	conv1_1	Convolution	224x224x64	3x3x3x64	1x1x64	1792
3	relu1_1	ReLU	224x224x64			
4	conv1_2	Convolution	224x224x64	3x3x64x64	1x1x64	36928
5	relu1_2	ReLU	224x224x64			
6	pool1	Max Pooling	112x112x64			
7	conv2_1	Convolution	112x112x128	3x3x64x128	1x1x128	73856
8	relu2_1	ReLU	112x112x128			
9	conv2_2	Convolution	112x112x128	3x3x128x128	1x1x128	147584
10	relu2_2	ReLU	112x112x128			
11	pool2	Max Pooling	56x56x128			
12	conv3_1	Convolution	56x56x256	3x3x128x256	1x1x256	295168
13	relu3_1	ReLU	56x56x256			
14	conv3_2	Convolution	56x56x256	3x3x256x256	1x1x256	590080
15	relu3_2	ReLU	56x56x256			
16	conv3_3	Convolution	56x56x256	3x3x256x256	1x1x256	590080
17	relu3_3	ReLU	56x56x256			
18	conv3_4	Convolution	56x56x256	3x3x256x256	1x1x256	590080
19	relu3_4	ReLU	56x56x256			
20	pool3	Max Pooling	28x28x256			
21	conv4_1	Convolution	28x28x512	3x3x256x512	1x1x512	1180160
22	relu4_1	ReLU	28x28x512			

23	conv4_2	Convolution	28x28x512	3x3x512x512	1x1x512	2359808
24	relu4_2	ReLU	28x28x512			
25	conv4_3	Convolution	28x28x512	3x3x512x512	1x1x512	2359808
26	relu4_3	ReLU	28x28x512			
27	conv4_4	Convolution	28x28x512	3x3x512x512	1x1x512	2359808
28	relu4_4	ReLU	28x28x512			
29	pool4	Max Pooling	14x14x512			
30	conv5_1	Convolution	14x14x512	3x3x512x512	1x1x512	2359808
31	relu5_1	ReLU	14x14x512			
32	conv5_2	Convolution	14x14x512	3x3x512x512	1x1x512	2359808
33	relu5_2	ReLU	14x14x512			
34	conv5_3	Convolution	14x14x512	3x3x512x512	1x1x512	2359808
35	relu5_3	ReLU	14x14x512			
36	conv5_4	Convolution	14x14x512	3x3x512x512	1x1x512	2359808
37	relu5_4	ReLU	14x14x512			
38	pool5	Max Pooling	7x7x512			
39	fc6	Fully Connected	1x1x4096	4096x25088	4096x1	102764544
40	relu6	ReLU	1x1x4096			
41	drop6	Dropout	1x1x4096			
42	fc7	Fully Connected	1x1x4096	4096x4096	4096x1	16781312
43	relu7	ReLU	1x1x4096			
44	drop7	Dropout	1x1x4096			
45	fc8	Fully Connected	1x1x1000	1000x4096	1000x1	4097000
46	Prob	Softmax	1x1x1000			
47	Output	Classification				

Inception V3: It is an advanced pre-trained CNN model. It consists of 316 layers and 350 connections. The number of convolution layers is 94 of different filter sizes, where the size of the first input layer is $299 \times 299 \times 3$. A brief description of this model is given in Table 2. In this table, it is shown that a scaling layer is added after the input layer. On the first convolution layer, activation is performed and obtained a weight matrix of dimension $149 \times 149 \times 32$, where 32 denotes the number of filters. Later, the batch normalization and ReLu activation layers are added. Mathematically, the ReLu layer is defined as:

$$\mathrm{Re}_i^{(l)} = \max\left(hv, hv_i^{(l-1)}\right) \tag{2}$$

Between the convolution layers, a pooling layer is also added to get active neurons. In the first max-pooling layer, the filter size is 2×2. Mathematically, max pooling is defined as:

$$mx_1^{(q)} = mx_1^{(q-1)} \tag{3}$$

$$mx_2^{(q)} = \frac{mx_2^{(q-1)} - F\left(q\right)}{S^q} + 1 \tag{4}$$

$$mx_3^{(q)} = \frac{mx_3^{(q-1)} - F\left(q\right)}{S^q} + 1 \tag{5}$$

Where, S^M denotes the stride, mx_1^M, mx_2^M, and mx_3^M are defined filters for feature set map such as 2×2, 3×3. Moreover, a few other layers are also added in this architecture, such as addition and concatenation layers. In the end, an average pool layer was added. The activation is performed, and in the output, a resultant weight matrix is obtained as a features map of dimension $1 \times 1 \times 2048$. The last layer is FC, and its learnable weight matrix is 1000×2048, and the ensuing feature matrix is $1 \times 1 \times 1000$. Mathematically, the FC layer is defined as follows:

$$Fc_i^{(l)} = f\left(z_i^{(l)}\right) with\ z_i^{(l)} = \sum_{j=1}^{n_1^{(l-1)}} \sum_{r=1}^{n_2^{(l-1)}} \sum_{s=1}^{n_3^{(l-1)}} w_{i,j,r,s}^{(l)} \left(Fc_i^{(l-1)}\right)_{r,s} \quad (6)$$

Table 2: Detailed description of Inception V3 pre-trained CNN model

S/N	Name	Type	Activation	Learnable			
				Weights	Bias	Offset	Scale
1	input_1	Image Input	299x299x3				
2	scaling	Scaling	299x299x3				
3	conv2d_1	Convolution	149x149x32	[3,3,3,32]	[1,1,32]		
4	batch_normalization_1	Batch Normalization	149x149x32			1x1x32	1x1x32
5	activation_1_relu	ReLU	149x149x32				
6	conv2d_2	Convolution	147x147x32	[3,3,32,32]	[1,1,32]		
7	batch_normalization_2	Batch Normalization	147x147x32			[1,1,32]	[1,1,32]
8	activation_2_relu	ReLU	147x147x32				
9	conv2d_3	Convolution	147x147x64	[3,3,32,64]	[1,1,64]		
10	batch_normalization_3	Batch Normalization	147x147x64			[1,1,64]	[1,1,64]
11	activation_3_relu	ReLU	147x147x64				
12	max_pooling2d_1	Max Pooling	73x73x64				
13	conv2d_4	Convolution	73x73x80	[1,1,64,80]	[1,1,80]		
14	batch_normalization_4	Batch Normalization	73x73x80			[1,1,80]	[1,1,80]
15	activation_4_relu	ReLU	73x73x80				
16	conv2d_5	Convolution	71x71x192	[3,3,80,192]	[1,1,192]		
17	batch_normalization_5	Batch Normalization	71x71x192			[1,1,192]	[1,1,192]
18	activation_5_relu	ReLU	71x71x192				
19	max_pooling2d_2	Max Pooling	35x35x192				
20	conv2d_9	Convolution	35x35x64	[1,1,192,64]	[1,1,64]		
21	batch_normalization_9	Batch Normalization	35x35x64			[1,1,64]	[1,1,64]
22	activation_9_relu	ReLU	35x35x64				
23	conv2d_7	Convolution	35x35x48	[1,1,192,48]	[1,1,48]		
24	conv2d_10	Convolution	35x35x96	[3,3,64,96]	[1,1,96]		
25	batch_normalization_7	Batch	35x35x48			[1,1,48]	[1,1,48]

		Normalization					
26	batch_normalization_10	Batch Normalization	35x35x96			[1,1,96]	[1,1,96]
27	activation_7_relu	ReLU	35x35x48				
28	activation_10_relu	ReLU	35x35x96				
29	average_pooling2d_1	Avg Pooling	35x35x192				
30	conv2d_6	Convolution	35x35x64	[1,1,192,64]	[1,1,64]		
31	conv2d_8	Convolution	35x35x64	[5,5,48,64]	[1,1,64]		
32	conv2d_11	Convolution	35x35x92	[3,3,96,96]	[1,1,96]		
33	conv2d_12	Convolution	35x35x32	[1,1,192,32]	[1,1,32]		
34	batch_normalization_6	Batch Normalization	35x35x64			[1,1,64]	[1,1,64]
35	batch_normalization_8	Batch Normalization	35x35x64			[1,1,64]	[1,1,64]
36	batch_normalization_11	Batch Normalization	35x35x96			[1,1,96]	[1,1,96]
37	batch_normalization_12	Batch Normalization	35x35x32			[1,1,32]	[1,1,32]
38	activation_6_relu	ReLU	35x35x64				
39	activation_8_relu	ReLU	35x35x64				
40	activation_11_relu	ReLU	35x35x96				
41	activation_12_relu	ReLU	35x35x32				
42	mixed0	Depth Concat	35x35x256				
43	conv2d_16	Convolution	35x35x64	[1,1,256,64]	[1,1,64]		
44	batch_normalization_16	Batch Normalization	35x35x64			[1,1,64]	[1,1,64]
45	activation_16_relu	Fully Connected	35x35x64				
46	conv2d_14	Convolution	35x35x48	[1,1,256,48]	[1,1,48]		
47	conv2d_17	Convolution	35x35x96	[3,3,64,96]	[1,1,96]		
--	--	--	--	--	--	--	--
307	batch_normalization_94	Batch Normalization	8x8x192			[1,1,192]	[1,1,192]
308	activation_86_relu	ReLU	8x8x320				
309	mixed9_1	Depth Concat	8x8x768				
310	concatenate_2	Depth Concat	8x8x768				
311	activation_94_relu	ReLU	8x8x192				
312	mixed10	Depth Concat	8x8x2048				
313	avg_pool	Avg Pooling	1x1x2048				
314	predictions	Fully Connected	1x1x1000	1000x2048	1000x1		
315	predictions_softmax	Softmax	1x1x1000				

316	Classification Layer_predictions	Classification Output					

Feature Extraction using TL: In the feature extraction step, we are employing TL. Using TL, we retrain both specific CNN models (VGG19 and InceptionV3) on the selected datasets. For training, we set a 60:40 approach along with labeled data. Furthermore, we perform preprocessing, in which we resize the images according to the input layer of each model. Later, we select the input convolutional layer and output layer as feature mapping. For VGG19, we choose the first convolutional layer as an input layer and FC7 as an output. After that, the CNN activation is performed, and we obtain training and testing vectors. On feature layer FC7, a resultant feature vector is obtained of dimension 1×4096 denoted by $\varphi^{(k1)}$ and utilized in the next process. A modified architecture of VGG19 is also shown in Figure 3. For Inception V3, we select the first convolutional layer as input and the Average Pool layer as a Feature Map. Similar to VGG19, we perform TL and retrain this model on the selected datasets, and apply the CNN activation on the Average Pool layer. On this layer, we obtain a feature vector of dimension 1×2048 and denoted by $\varphi^{(k2)}$. Both training and testing vectors proceed for the next features fusion process. The modified architecture of Inception V3 is shown in Figure 2. In this figure, it is shown that the last three layers are removed before retrained on the selected datasets for this work.

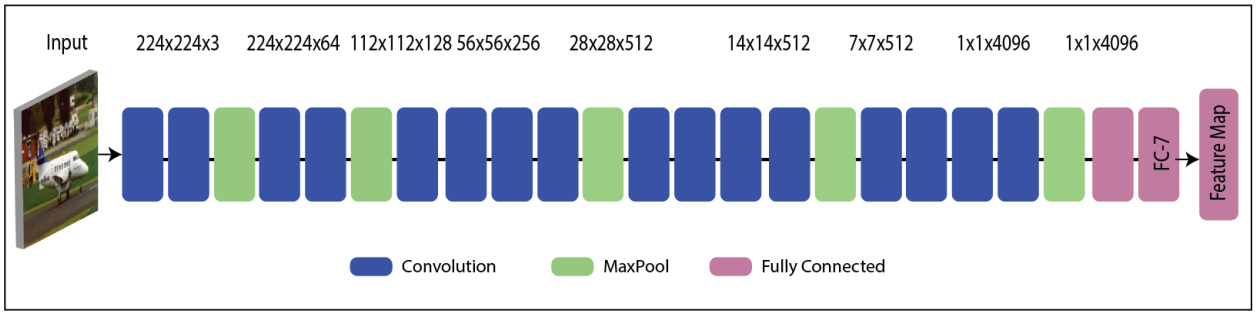

Figure 2: *Modified VGG-19 architecture for features extraction*

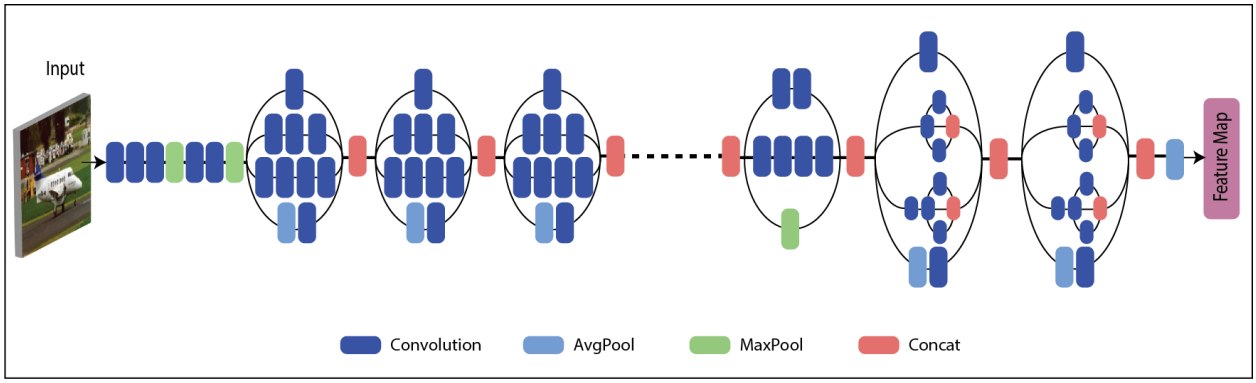

Figure 3: *Modified Inceptionv3 architecture for features extraction*

4.2. Feature Fusion

The fusion of multiple features in one matrix is the latest research area of pattern recognition. The primary purpose of feature fusion is to obtain a stronger feature vector for classification. From the latest research, it is noticed that the fusion process improves the overall accuracy, but on the other side, its main disadvantage is high computational time (sec). However, our usual priority is to improve the classification accuracy. For this purpose, we implement a new Parallel Maximum Covariance (PMC) approach for features fusion. In this approach, we need to equalize the length of both extracted feature vectors. Later, we find the maximum covariance for fusion in a single matrix.

Consider two deep learning feature vectors $\varphi^{(k1)}$ and $\varphi^{(k2)}$ of dimensions $n \times m$ and $n \times q$, where n denotes the number of images, m indicates VGG19 deep learning feature vector length of $n \times 4096$ and q denotes Inception V3 feature vector of dimension $n \times 2048$, respectively. To make the length of vectors equal, we first find out the maximum length vector and perform average value padding. The average feature is calculated from a higher length vector. Let $\boldsymbol{a}$ be an arbitrary unit column m vector presenting a pattern in φ_1 field and $\boldsymbol{b}$ indicates a random unit column vector representing a pattern in the φ_2 field, respectively. For time series projection on row vectors are defined as follows:

$$x_1 = \varphi_1^T \, \varphi^{(k1)} \tag{7}$$

$$x_2 = \varphi_2^T \, \varphi^{(k2)} \tag{8}$$

For optimal solutions φ_1 and φ_2, maximize their covariance as follows:

$$\tilde{c} = Cov[x_1, x_2] \tag{9}$$

$$\tilde{c} = Cov\left[\varphi_1^T \, \varphi^{(k1)}, \varphi_2^T \, \varphi^{(k2)}\right] \tag{10}$$

$$\tilde{c} = \frac{1}{n-1}\left[\varphi_1^T \, \varphi^{(k1)}\left(\varphi_2^T \, \varphi^{(k2)}\right)\right] \tag{11}$$

$$\tilde{c} = \varphi_1\left(\boldsymbol{C}_{\varphi_1\varphi_2}\right)\varphi_2 \tag{12}$$

$$\boldsymbol{C}_{\varphi_1\varphi_2} = \frac{1}{n-1}\left(\varphi^{(k1)} \, \varphi^{(k2)^T}\right) \tag{13}$$

Where, $\boldsymbol{C}_{\varphi_1\varphi_2}$ is the covariance value among φ_1 and φ_2 whose ith and jth features are $\varphi_i(t)$ and $\varphi_j(t)$. Hence, the feature pair i and j of maximum covariance $\boldsymbol{C}_{\varphi_1\varphi_2}$ is saved in the final fused vector. However, it is possible that few of the feature pairs are redundant. This process is continued until all pairs are compared with each other. In the end, a fused vector is obtained denoted by $\varphi^{(fu)}$ of dimensions $N \times K$ where K denotes the feature-length, which varies based on the selected features. In this work, the fused feature-length is $N \times 3294$ for the Caltech101 dataset, $N \times 2981$ for Birds dataset, and $N \times 3089$ for Butterflies dataset.

4.3. Feature Selection

Feature selection is an exciting research topic in ML nowadays and shows significant classification performance. In this work, we propose a new technique for feature selection; namely, Multi Logistic Regression controlled Entropy-Variances (MRcEV). A partial derivative-based activation function is used to remove the irrelevant features, and remaining robust features are passed to Entropy-variances function. Through the latter, a new vector is obtained, which only contains positive values. Finally, this vector is presented to ESD fitness function, and the validity of the proposed technique is determined. Mathematically, the formulation is given as:

For a given dataset, a fused vector is represented as $\Delta = \left\{\varphi^{(fu)}, y^{(fu)}\right\}_{fu=1}^{N}$ having N sample images, where $\varphi^{(fu)}$ denotes fused feature vector which is utilized as input and $\varphi^{(fu)} \in \mathbb{R}^p$. The $y^{(fu)}$ indicates corresponding labels and defined as $y^{(fu)} \in \mathbb{R}$. The probability among $\varphi^{(fu)}$ for the class i is then computed as follows:

$$p\left(y^{(fu)}|\varphi^{(fu)}\right) = \frac{\exp\left\{r_i^{(fu)}\right\}}{\sum_{j=1}^{q} exp\left\{r_i^{(fu)}\right\}} \tag{14}$$

$$r_i^{(fu)} = \sum_{j=1}^{p} \beta_{ij} \varphi_j^{(fu)} \tag{15}$$

The parameter of logistic regression $r_i = (r_0, r_1, \ldots, r_p)$ is obtained by minimizing the negative likelihood of features. If features are independent, then multinomial distribution is computed as follows:

$$E_\Delta = -\sum_{fu}^{n} \sum_{i=1}^{q} y_i^{(fu)} \log \, p\left(y^{(fu)}|\varphi^{(fu)}\right) \tag{16}$$

To get a sparse model, a regularization parameter $\tilde{\beta}$ is added to negative log-likelihood. The modified MLR criteria for active features are defined as follows:

$$M = E_\Delta + \tilde{\beta} E_r \tag{17}$$

$$E_r = \sum_{i=1}^{p} |r_i|, \text{ where } r_i \text{ is regularization parameter} \quad (18)$$

At the minimum value of M, the partial derivative $w.r.t\ r_i$ is formulated as follows:

$$\begin{cases} \left|\frac{\partial E_\Delta}{\partial r_i}\right| = \tilde{\beta} & if \ \ |r_i| > 0 \\ \left|\frac{\partial E_\Delta}{\partial r_i}\right| < \tilde{\beta} & if \ \ |r_i| = 0 \end{cases} \quad (19)$$

This expression shows that if the partial derivative of E_Δ $w.r.t\ r_i$ is less then $\tilde{\beta}$, then that feature value is set to zero, and removed from the final vector. Later, Entropy-Variances based new function is implemented to obtain a more robust vector. Mathematically, this function is formulated as:

$$Ent\left(\lambda(FV)\right) = -\left(\frac{\ln\left(cd\left(i+1\right)+\sigma^2\right)}{\ln\left(cd_{(i)}+\sigma^2\right)+\ln\left(cd_{(i)}-\sigma^2\right)} \right) \quad (20)$$

The selected features are passed to this function to get a clear difference among all features. This proposed selection technique picks almost 50% to 60% robust features from the fused feature vector. The selected features are finally verified through ESD classifier [36]. In the ensemble learning classifier, the subspace discriminant method is used. The proposed system predicted results are shown in Figure 5, 6, and 7.

Figure 4: Proposed system predicted labeled output for Caltech-101 dataset

Figure 5: Proposed system predicted labeled output for Birds dataset

Figure 6: *Proposed system predicted labeled output for Butterflies dataset.*

5. Results

This section presents the simulation results with detailed numerical analysis and visual plots. Three datasets are used for evaluation of the proposed method, such as Caltech101, Birds database, and Butterflies database. The Caltech101 [37] is a challenging dataset for object classification. It contains a total of 9144 images of 101 different object classes. The Birds database [38] is another publicly available dataset used for the challenge of object recognition. It contains a total of 600 images of six different types of bird categories. The Butterflies [39] database is another publicly available dataset, and it consists of a total of seven different classes. The total number of images in this dataset is 619, and each class possesses 42-134 images. A brief description of the selected datasets is also given in Table 3. For validation, the 60:40 approach is employed along with 10-Fold cross-validation. Many classifiers are using for the experimental process, such as Ensemble learning, SVM, KNN, and Linear Discriminant. The performance of each classifier is validated using three essential measures, including accuracy, FNR, and computational time. All the simulations were conducted in MATLAB2019a installed on a 2.4 Gigahertz Corei7 processor with 16 Gigabytes of RAM, 128 SSD, and a Radeon R7 graphic card.

Table 3: Numerical description of selected datasets

Image Database	Sample Classes	Total Samples	Min-Max
Caltech [37]	101	9144	31~800
Birds [38]	6	600	100~100
Butterflies [39]	7	619	42~134

5.1. Caltech-101 Dataset Results

The results achieved on the Caltech-101 dataset are discussed in this section. The results are presented in three different ways: In the first method, both VGG19 and inceptionV3 based deep features are fused using a serial-based method, and the classification is performed. In this method, no feature selection is performed. In the second method, the fusion of deep features is conducted using the proposed fusion approach, as presented in Section 4.2. In the third method, the feature selection is performed on the proposed fused vector, followed by classification. The results are

shown in Table 4, where it is evident that the ESD classifier gives better results as compared to other listed classifiers against each method. However, it may be noticed that a massive difference exists among the accuracies achieved against M1 and other methods. For the ESD classifier, the accuracy of M1 is 79%, but after applying P-Fusion method, the efficiency is improved, and achieved accuracy is 90.8%. Moreover, the computational time is also minimized by using P-Fusion method. On ESD classifier, the computational time was 120 (sec) for M1, whereas after applying P-Fusion, the time is decreased to 93. 70 (sec). By using the P-Selection method, the achieved accuracy is 95.5%, and the computational time is 47.0 (sec), which is better as compared to M1 and P-Fusion methods. Moreover, the accuracy of the P-Selection method is also verified through Figure 7. A comparison of classification accuracy between different classifiers is also conducted and shown in Table 4. It may be observed that the maximum accuracies achieved by ES-KNN, LDA, LSVM, QSVM, Cu-SVM, FKNN, MKNN, WKNN, and Co-KNN are 85.3%, 94.4%, 91.6%, 92%, 92.3%, 89.9%, 89.6%, 90.5%, and 92.8% respectively. Additionally, the computational time of Co-KNN classifier is better as compared to other methods. Overall, the proposed selection method shows significant performance on ESD classifier for the Caltech101 dataset.

Table 4: *Proposed classification results using the Caltech101 dataset. M1 represents simple serial-based fusion and classification, P-Fusion represents the proposed fusion approach, and P-Selection represents the proposed selection method results.*

Classifier	M1	P-Fusion	P-Selection	Accuracy (%)	FNR (%)	Time (s)
ESD	✓			**79.0**	**21.0**	180.00
		✓		**90.8**	**9.2**	93.70
			✓	**95.5**	**4.5**	47.00
ES-KNN	✓			75.8	24.2	665.80
		✓		80.1	19.9	286.45
			✓	85.3	14.7	191.27
LDA	✓			75.0	25.0	597.84
		✓		81.8	18.2	127.83
			✓	94.4	5.5	106.57
L-SVM	✓			76.0	24.0	9723.70
		✓		88.0	12.0	3154.70
			✓	91.6	8.6	2045.00
Q-SVM	✓			77.2	22.8	1896.00
		✓		87.6	12.4	1341.00
			✓	92.0	8.0	753.57
Cu-SVM	✓			77.9	22.1	7493.00
		✓		87.7	12.3	3647.70
			✓	92.3	7.7	1889.50
F-KNN	✓			75.7	24.3	152.06
		✓		84.9	15.1	96.96
			✓	89.9	10.1	71.57

	✓			74.8	25.2	**57.95**
M-KNN		✓		84.5	15.5	47.44
			✓	89.6	10.4	33.90
	✓			76.8	23.2	228.19
W-KNN		✓		85.7	14.3	187.50
			✓	90.5	9.5	105.87
	✓			52.4	21.0	61.35
Co-KNN		✓		87.6	12.4	48.76
			✓	92.8	7.2	23.83

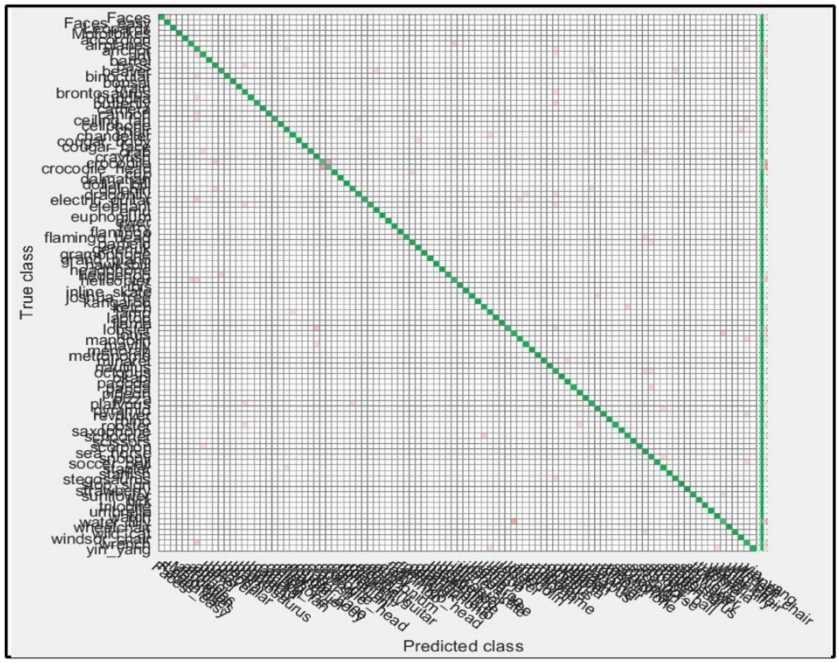

Figure 7: Confusion matrix of the proposed selection accuracy on ESD classifier

5.2. Birds Dataset Results

The classification results using birds dataset are presented in this section. Similar to the Caltech101 dataset, three methods are also applied for the evaluation of this dataset. The results are given in Table 5. In this table, it is shown that the ESD classifier gives better results for all three methods as compared to other listed classifiers. The ESD classifier achieves an accuracy of 99% against M1, but after applying the P-Fusion method, the efficiency is improved, and the accuracy is increased to 99.5%. Moreover, the computational time is also minimized by using the P-Fusion method. The computational time for M1 is 85.09 (sec) by the ESD, which reduces to 68.31 (sec) after applying the P-Fusion. After using the proposed selection method, both accuracy and computational times are improved. By using P-Selection method, the achieved accuracy is maximum (100%), and computational time is 42.45 (sec), which is better as compared to M1 and P-Fusion methods. Moreover, the accuracy of P-Selection method is also verified through Figure 8. Furthermore, the classification based comparison is conducted with other classifiers, as listed in Table 5. In this table, it is shown that all the different classifiers also achieve their maximum accuracies for the proposed

selection method. Additionally, the computational time of the W-KNN classifier is better as compared to other methods. The noted time of W-KNN for all the three methods is 23.96 (sec), 13.10 (sec), and 9.16 (sec) respectively. However, overall the ESD classifier shows sufficiently increased accuracy using the P-Selection method as well as the P-Fusion.

Table 5: Proposed classification results using the Birds dataset. ***M1*** *represents simple serial-based fusion and classification,* ***P-Fusion*** *represents the proposed fusion approach, and* ***P-Selection*** *represents the proposed selection method results.*

Classifier	M1	P-Fusion	P-Selection	Accuracy (%)	FNR (%)	Time (s)
ESD	✓			**99.0**	15.5	85.09
		✓		**99.5**	1.0	68.31
			✓	**100.0**	0.0	42.45
E-S-KNN	✓			96.7	3.3	45.09
		✓		97.6	2.4	38.31
			✓	97.4	2.6	25.54
LD	✓			98.0	2.0	48.39
		✓		99.0	1.0	31.11
			✓	100.0	0.0	23.92
L-SVM	✓			97.9	2.1	45.36
		✓		99.0	0.5	20.00
			✓	100.0	0.0	17.66
Q-SVM	✓			84.5	1.0	51.03
		✓		99.3	0.7	24.06
			✓	100.0	0.0	15.25
Cub-SVM	✓			99.0	1.0	54.59
		✓		99.5	0.5	43.32
			✓	100.0	0.0	21.29
F-KNN	✓			96.2	3.8	41.47
		✓		97.4	2.6	19.58
			✓	99.5	0.5	14.89
M-KNN	✓			97.6	2.4	32.30
		✓		98.8	1.2	17.31
			✓	100.0	0.0	15.82
W-KNN	✓			97.9	2.1	23.96
		✓		99.3	0.7	13.10
			✓	100.0	0.0	9.16
Cos-KNN	✓			95.7	4.3	31.08
		✓		99.0	1.0	22.00
			✓	99.8	0.2	16.11

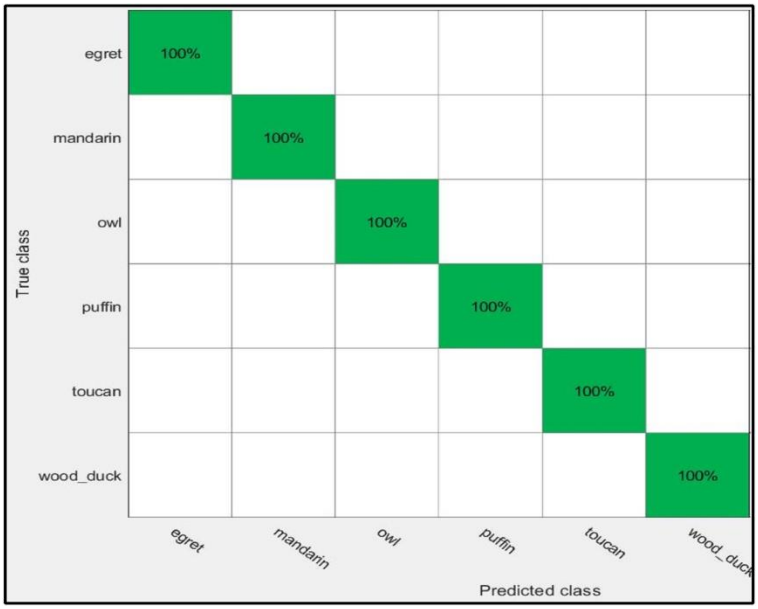

Figure 8: Confusion Matrix for Birds dataset using proposed selection method on ESD classifier

5.3. Butterflies Dataset

In this section, classification results are presented for the butterfly dataset. As mentioned above, this dataset includes a total of 7 different butterfly types. Results are presented using three different methods, as discussed in Section 5.3. The results are given in Table 6. It may be observed that the ESD classifier gives better outcomes for all three feature methods, such as M1, P-Fusion, and P-selection. For M1, the ESD classifier achieves an accuracy of 95.1%, which is improved to 95.6% after using the P-Fusion method. The computational time of M1 is 46.05 (sec), but after P-Fusion, the time is reduced to 31.95 (sec). In comparison, the P-Selection method achieves an accuracy of 98%, which is better than the M1 and P-Fusion. Moreover, the computational time of this method is 19.53 (sec), which is also the minimum. The performance of the ESD classifier for the P-Selection method may also be verified through Figure 9. The performance of the ESD classifier is also compared with a few other well-known techniques such as SVM, KNN, and LDA, as given in Table 6. From results, it can be clearly seen that all the classifiers provide better accuracy on the P-Selection method. Moreover, it is also concluded that the W-KNN performs better in terms of the computational time.

Table 6: *Proposed classification results using the Butterflies dataset.* ***M1*** *represents simple serial-based fusion and classification,* ***P-Fusion*** *represents the proposed fusion approach, and* ***P-Selection*** *represents the proposed selection method results.*

Classifier	M1	P-Fusion	P-Selection	Accuracy (%)	FNR (%)	Time (s)
ESD	✓			**95.1**	9.4	46.05
		✓		**95.6**	5.9	31.95
			✓	**98.0**	2.0	19.53
E-S-KNN	✓			85.7	14.3	28.56
		✓		87.7	12.3	18.27
			✓	88.7	11.3	13.08

	✓			70.9	29.1	48.44
LD		✓		94.1	4.6	22.42
			✓	96.6	3.4	17.01
	✓			91.6	8.4	40.02
L-SVM		✓		94.6	5.4	29.65
			✓	96.6	3.4	16.72
	✓			94.1	5.9	39.46
Q-SVM		✓		94.1	5.9	24.58
			✓	96.6	3.4	18.80
	✓			90.6	4.9	44.23
Cub-SVM		✓		93.6	6.4	29.41
			✓	97.0	3.0	21.51
	✓			85.7	14.3	30.82
F-KNN		✓		89.2	10.8	18.70
			✓	94.1	5.9	13.79
	✓			82.3	19.7	29.29
M-KNN		✓		85.2	14.8	18.30
			✓	92.1	7.9	10.83
	✓			85.2	14.8	**15.06**
W-KNN		✓		87.2	12.8	**14.26**
			✓	94.6	5.4	**10.12**
	✓			81.8	18.2	16.02
Cos-KNN		✓		85.7	14.3	14.54
			✓	94.1	5.9	10.55

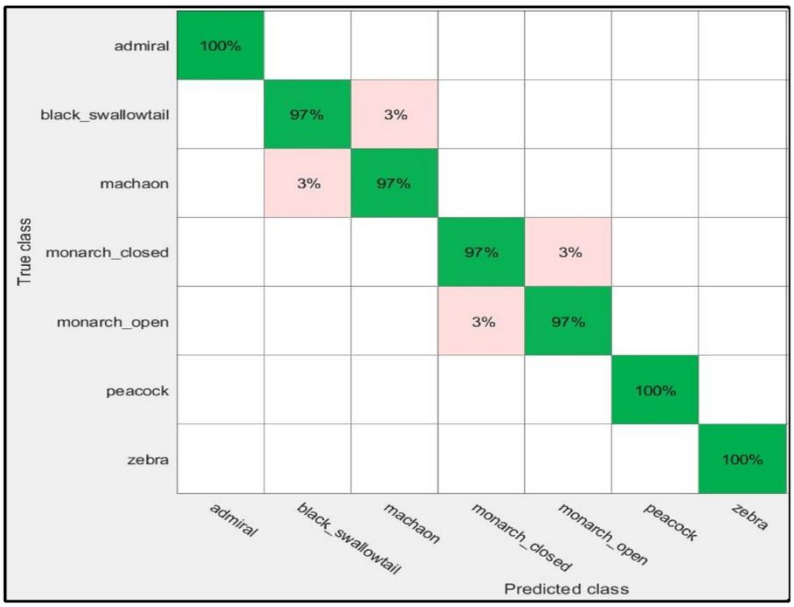

Figure 9: *Confusion Matrix for Butterflies dataset*

5.4. Analysis and Comparison with Existing Techniques

A comprehensive analysis and comparison with existing techniques are presenting in this section to examine the authenticity of the proposed method results. As mentioned, three datasets are using for validation, such as Caltech101, Birds database, and Butterfly database. The proposed fusion and robust feature selection methods give a significant performance of 95.5%, 100%, and 98%, respectively, for ESD classifier on the selected datasets. However, it is essential to examine the accuracy of ESD on each classifier based on a detailed statistical analysis. For Caltech101 dataset, we run the proposed algorithm 500 times for each method and get two accuracies- average (76.3%, 87.9%, and 92.7%), and maximum (79%, 90.8%, and 95.5%). These accuracies are also plotted in Figure 10 (a). In this figure, it is shown that a minor change is occurring in the accuracy after 500 iterations. For Birds database, two accuracies are also obtained – minimum (97.2%, 98.9%, and 99.4%) and maximum (99%, 99.5%, and 100%). These values are also plotted in Figure 10 (b). In this figure, it can be observed that the change in M1 is a bit higher as compared to P-Fusion and P-Selection. In last, the statistical analysis is conducting for butterflies dataset, as showing in Figure 10 (c). In this figure, it is shown that a bit of change in the accuracy of each method is observed. Overall, it can be proved the authenticity of the proposed fusion and selection methods. A brief comparison with existing techniques is also presented in Table 7. From this table, it is evident that the proposed method gives improved accuracy.

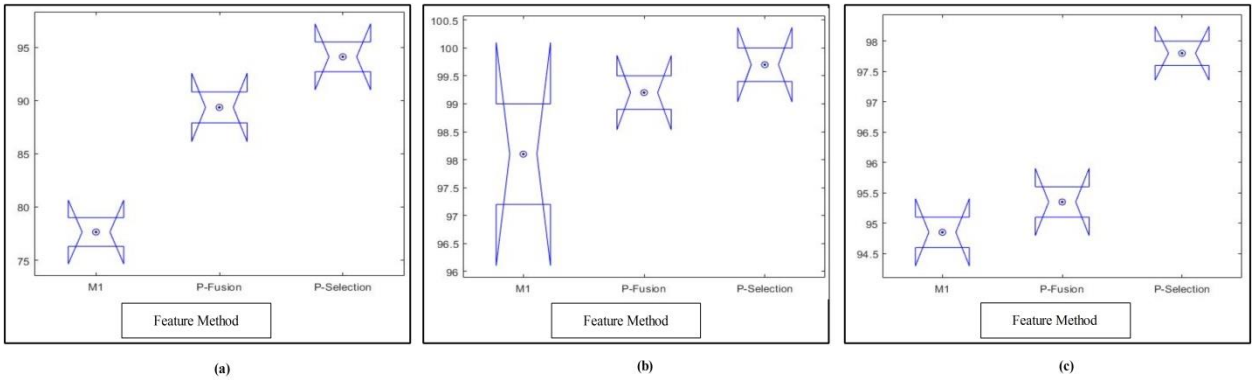

Figure 10: Statistical analysis of ESD classifier using all three methods

Table 7: *Comparison of proposed accuracy with recent techniques, MLFFS means "Multi-Layers Features Fusion and Selection."*

Reference	Technique	Dataset	Accuracy (%)
Roshan et al. [22]	Fine-tuning on top layers	Caltech101	91.66
Jongbin et al. [23]	Discrete Fourier transform	Caltech101	93.6
Qun et al. [24]	Memory Banks based unsupervised learning	Caltech101	91.0
Qing et al. [25]	PCA based reduction on fused features	Caltech101	92.54
Xueliang et	A fusion of Mid-Level	Caltech101	92.2

al. [26]	layers-based features		
Rashid et al. [31]	Fusion of SIFT and CNN features	Caltech101	89.7
Svetlana [39]	Local affine parts based approach	Butterflies	90.4
Proposed	**MLFFS**	Butterflies	**98.0**
Proposed	**MLFFS**	Birds	**100%**
Proposed	**MLFFS**	Caltech101	**95.5**

5. Conclusions

A new multi-layer deep features fusion and selection based method for object classification is presented in this work. The major contribution of this work lies in the fusion of deep learning models and then selection of the robust features for final classification. Three core steps are involved in the proposed system: features extraction using transfer learning, features fusion of two different deep learning models (VGG19 and Inception V3) using PMC, and selection of the robust features using Multi Logistic Regression controlled Entropy-Variances (MRcEV) method. An ESDA classifier is used to validate the performance of MRcEV. We utilize three datasets for the experimental process, and demonstrate improved achieved accuracy. From the results, we conclude that the proposed method is useful for large datasets as well as small datasets. Fusion of two different deep learning features shows an impact on the classification accuracy. Also, the selection of robust features shows an effect on both computational time and classification accuracy. The main limitation of the proposed method is the quality of features. By using low-quality images, it is not possible to get strong features. In future, this problem will be rectified through contrast stretching deep learning architecture. Moreover, for the experimental process, Caltech256 dataset will also be considered.

Author Contributions: Authors M. Rashid and MA. Khan develop this idea and responsible for first draft. Author, M.AH responsible for mathematical formulation, Dr. S.H Wang supervised this work, Dr. R. Naqvi gives technical support of this work, Dr. Tanzila and Dr. A. Rehman responsible for final proofreading.

Funding: There is no funding involves for this work.

Acknowledgments: N/A

Conflicts of Interest: "The authors declare no conflict of interest."

References

1. Ly, H.-B., et al., *Computational Hybrid Machine Learning Based Prediction of Shear Capacity for Steel Fiber Reinforced Concrete Beams.* Sustainability, 2020. **12**(7): p. 2709.
2. Cioffi, R., et al., *Artificial Intelligence and Machine Learning Applications in Smart Production: Progress, Trends, and Directions.* Sustainability, 2020. **12**(2): p. 492.
3. Lin, F., et al., *Detection of corn and weed species by the combination of spectral, shape and textural features.* Sustainability, 2017. **9**(8): p. 1335.

4. Zhou, C., et al., *An Improved Style Transfer Algorithm Using Feedforward Neural Network for Real-Time Image Conversion.* Sustainability, 2019. **11**(20): p. 5673.
5. Amini, M.H., H. Arasteh, and P. Siano, *Sustainable smart cities through the lens of complex interdependent infrastructures: panorama and state-of-the-art*, in *Sustainable interdependent networks II*. 2019, Springer. p. 45-68.
6. Gupta, V. and J. Singh, *Study and Analysis of Back-Propagation Approach in Artificial Neural Network Using HOG Descriptor for Real-Time Object Classification*, in *Soft Computing: Theories and Applications*. 2019, Springer. p. 45-52.
7. Sharif, M., et al., *Deep CNN and geometric features-based gastrointestinal tract diseases detection and classification from wireless capsule endoscopy images.* Journal of Experimental & Theoretical Artificial Intelligence, 2019: p. 1-23.
8. Rashid, M., et al., *Object detection and classification: a joint selection and fusion strategy of deep convolutional neural network and SIFT point features.* Multimedia Tools and Applications, 2018: p. 1-27.
9. Wang, S., et al., *An Improved Difference of Gaussian Filter in Face Recognition.* Journal of Multimedia, 2012. **7**(6): p. 429-433.
10. He, Q., et al., *Multimedia based fast face recognition algorithm of speed up robust features.* Multimedia Tools and Applications, 2019: p. 1-11.
11. Suhas, M. and B. Swathi, *Significance of Haralick Features in Bone Tumor Classification Using Support Vector Machine*, in *Engineering Vibration, Communication and Information Processing*. 2019, Springer. p. 349-361.
12. Khan, M.A., et al., *Construction of saliency map and hybrid set of features for efficient segmentation and classification of skin lesion.* Microscopy research and technique, 2019.
13. Szegedy, C., et al., *Intriguing properties of neural networks.* arXiv preprint arXiv:1312.6199, 2013.
14. Szegedy, C., et al. *Going deeper with convolutions*. in *Proceedings of the IEEE conference on computer vision and pattern recognition*. 2015.
15. He, K., et al. *Deep residual learning for image recognition*. in *Proceedings of the IEEE conference on computer vision and pattern recognition*. 2016.
16. Szegedy, C., et al. *Inception-v4, inception-resnet and the impact of residual connections on learning*. in *Thirty-First AAAI Conference on Artificial Intelligence*. 2017.
17. Jiang, B., et al., *Probabilistic Feature Selection and Classification Vector Machine.* ACM Transactions on Knowledge Discovery from Data (TKDD), 2019. **13**(2): p. 21.
18. Xiao, X., et al., *A feature extraction method for lung nodules based on a multichannel principal component analysis network (PCANet).* Multimedia Tools and Applications, 2019: p. 1-19.
19. Wen, J., et al., *Robust sparse linear discriminant analysis.* IEEE Transactions on Circuits and Systems for Video Technology, 2019. **29**(2): p. 390-403.
20. Mwangi, B., T.S. Tian, and J.C. Soares, *A review of feature reduction techniques in neuroimaging.* Neuroinformatics, 2014. **12**(2): p. 229-244.
21. Khan, M.A., et al., *An implementation of normal distribution based segmentation and entropy controlled features selection for skin lesion detection and classification.* BMC cancer, 2018. **18**(1): p. 638.
22. Gopalakrishnan, R., Y. Chua, and L.R. Iyer, *Classifying neuromorphic data using a deep learning framework for image classification.* arXiv preprint arXiv:1807.00578, 2018.
23. Ryu, J., M.-H. Yang, and J. Lim. *DFT-based Transformation Invariant Pooling Layer for Visual Classification*. in *Proceedings of the European Conference on Computer Vision (ECCV)*. 2018.

24. Liu, Q. and S. Mukhopadhyay, *Unsupervised Learning using Pretrained CNN and Associative Memory Bank.* arXiv preprint arXiv:1805.01033, 2018.
25. Li, Q., Q. Peng, and C. Yan, *Multiple VLAD encoding of CNNs for image classification.* Computing in Science & Engineering, 2018. **20**(2): p. 52-63.
26. Liu, X., et al., *On fusing the latent deep CNN feature for image classification.* World Wide Web, 2018: p. 1-14.
27. Khan, H.A., *DM-L Based Feature Extraction and Classifier Ensemble for Object Recognition.* Journal of Signal and Information Processing, 2018. **9**(02): p. 92.
28. Mahmood, A., et al. *Resfeats: Residual network based features for image classification*. in *Image Processing (ICIP), 2017 IEEE International Conference on*. 2017. IEEE.
29. Cengil, E., A. Çınar, and E. Özbay. *Image classification with caffe deep learning framework*. in *Computer Science and Engineering (UBMK), 2017 International Conference on*. 2017. IEEE.
30. Zhang, C., Q. Huang, and Q. Tian, *Contextual Exemplar Classifier-Based Image Representation for Classification.* IEEE Transactions on Circuits and Systems for Video Technology, 2017. **27**(8): p. 1691-1699.
31. Rashid, M., et al., *Object detection and classification: a joint selection and fusion strategy of deep convolutional neural network and SIFT point features.* Multimedia Tools and Applications, 2019. **78**(12): p. 15751-15777.
32. Hussain, N., et al., *A deep neural network and classical features based scheme for objects recognition: an application for machine inspection.* Multimedia Tools and Applications, 2020.
33. Khan, M.A., et al., *An implementation of optimized framework for action classification using multilayers neural network on selected fused features.* Pattern Analysis and Applications, 2018: p. 1-21.
34. Liaqat, A., et al., *Automated ulcer and bleeding classification from WCE images using multiple features fusion and selection.* Journal of Mechanics in Medicine and Biology, 2018. **18**(04): p. 1850038.
35. Simonyan, K. and A. Zisserman, *Very deep convolutional networks for large-scale image recognition.* arXiv preprint arXiv:1409.1556, 2014.
36. Gomes, H.M., et al., *A survey on ensemble learning for data stream classification.* ACM Computing Surveys (CSUR), 2017. **50**(2): p. 1-36.
37. Fei-Fei, L., R. Fergus, and P. Perona, *One-shot learning of object categories.* IEEE transactions on pattern analysis and machine intelligence, 2006. **28**(4): p. 594-611.
38. Lazebnik, S., C. Schmid, and J. Ponce. *A maximum entropy framework for part-based texture and object recognition*. in *Computer Vision, 2005. ICCV 2005. Tenth IEEE International Conference on*. 2005. IEEE.
39. Lazebnik, S., C. Schmid, and J. Ponce. *Semi-local affine parts for object recognition*. in *British Machine Vision Conference (BMVC'04)*. 2004. The British Machine Vision Association (BMVA).