Received September 10, 2019, accepted September 29, 2019, date of publication October 21, 2019,
date of current version October 30, 2019.

Digital Object Identifier 10.1109/ACCESS.2019.2947717

An Efficient Framework for Animal Breeds Classification Using Semi-Supervised Learning and Multi-Part Convolutional Neural Network (MP-CNN)

S. DIVYA MEENA AND L. AGILANDEESWARI
School of Information Technology and Engineering, Vellore Institute of Technology, Vellore 632014, India

Corresponding author: L. Agilandeeswari (agila.l@vit.ac.in)

ABSTRACT The automatic classification of animal images is an onerous task due to the challenging image conditions, especially when it comes to animal breeds. In this paper, we built a semi-supervised learning based Multi-part Convolutional Neural Network (MP-CNN) that classifies 35,992 animal images from ImageNet into 27 different classes of animals. The proposed model classifies the animals on both generic and fine-grained level. The animal breeds are accurately classified using Multi-part Convolutional Neural Network with a hybrid feature extraction framework of Fisher Vector based Stacked Autoencoder. Furthermore, with Semi-supervised learning based pseudo-labels, the model classifies new classes of unlabeled images too. Modified Hellinger Kernel classifier has been used to re-train the misclassified classes of animals and thereby improve the performance obtained from MP-CNN. The model has experimented with varied tasks to analyze its performance in each of the cases. The experimental results have proved that the coalesced approach of MP-CNN with pseudo-labels can accurately classify animal breeds and we have achieved an accuracy of 99.95% from the proposed model.

INDEX TERMS Fisher vector, inception-V3, modified hellinger Kernel classifier, multi-part based convolutional neural network, pseudo-labels, semi-supervised learning, stacked autoencoder.

I. INTRODUCTION

Despite being the oldest computing technique, image classification remains an indispensable one. It has come a long way from using Fourier transforms to using neural networks. However, it remains a complicated computation because of the challenges in the images such as pose variations, occlusion, illumination, camouflage and more. With Deep learning, one can make a system to perform classification on its own [1]. Deep learning, a kind of machine learning lets the model perform classification directly from the training source like images, text, or sound. This requires the construction of a Deep Neural Network (DNN). Building a model from scratch may have better performance but it is quite complicated and time-consuming too. Instead, one can use the concept of Transfer Learning to build very efficient neural networks.

While image classification is used almost in all aspects, its use is not fully accomplished in certain fields. One such field is the classification of animal species. The Automatic classification of animal images remains an unsolved problem due to the challenges in images. When it comes to image classification and recognition, animals are the difficult ones [2]. Deep learning can aid in such scenarios. It provides a wide range of powerful algorithms with which the whole process can be simplified and automated [3].

Fine-Grained Classification (FGC) is a sub-field of generic image classification, where the main objective is to discriminate the secondary level detail of an image within the primary level. Classifying an animal as a dog or cat is primary level classification and classifying the dog either as Poodle or Pug is secondary level classification. FGC of animals is quite tedious because of the huge intra-class variation in the sub-categories and little inter-class variation among the various sub-categories. This complexity arises due to the visual and semantic similarities of the animals in the sub-categories.

The associate editor coordinating the review of this manuscript and approving it for publication was Zhanyu Ma.

The sub-categories of animals have a subtle variation, which is further complicated when the image has challenges like scaling, rotation, variation in posture, camouflage or occlusion. An illustration of generic vs. fine-grained classification is given in Fig. 1.

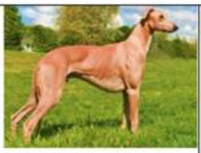
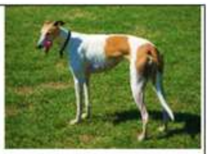
Input image		
Generic classification	Dog	Dog
Fine-grained classification	greyhound	Whippet

FIGURE 1. Illustration of generic vs. fine-grained classification.

Most of the literature on FGC, have either used annotations for either objects or the parts for extracting the discriminative features. Hierarchical Part Matching (HPM) [4], Part-based One-vs.-One Features (POOF) [5], Pose Normalized Deep Convolutional Nets (PN-DCN) [6], and Part based R-CNN [7] utilizes annotations on both object and part level for both train and test data. However, annotations are expensive and human-intensive. In particular, part annotations are tedious and prone to error. Deep Localization, Alignment and Classification (Deep-LAC) [8], and Part-Stacked CNN (PS-CNN) [9] proposed models for fine-grained visual categorization using annotations for object localization in both train and test images. Coarse-to-fine [10], Webly-supervised [11], and PG Alignment [12] used hand-crafted SIFT features and annotations on the object level.

Xie et al. proposed InterActive [13], a novel algorithm that measures the attention of the neurons and improves the low-level neurons to improve the classification performance. TL Atten [14] ignored the spatial relationship between the object-part and among the parts. Simon and Rodner [15] proposed a Constellation of Neural Activations (CAN) to localize the discriminative parts without using the bounding box. However, they have not localized the object in the first place. Lin *et al.* [16] proposed Bilinear CNN that combines two CNN for extracting features. The model, however, does not use any annotations like Fused One-vs-All Features (FOAF) [17] and Dense Graph Matching (DGM) [18].

Recently, Zheng *et al.* [19] proposed a novel Multi-Attention Convolutional Neural Network (MA-CNN) for fine-grained classification problems. The network consists of convolutional layers, channel grouping and a sub-network for part classification. One other common problem found in existing FGC works is that they do not consider the relationship among the parts and the relationship between the features extracted by different parts [20]. In yet another case, when the object is localized before choosing the parts, then the spatial relationship between the object and the parts should not be ignored [21]. The spatial relationship among the parts and between the object and the parts are highly useful in extracting discriminating parts, which makes the classification easier.

Generally, when the objects are localized before choosing the discriminative parts, we found two common issues. The first one is that the localized objects contain a larger background than the size of the object. The second problem is that the background and the object have a large overlap, which leads to redundant information. To overcome the above said shortcomings, we propose a novel Multi Part Convolutional Neural Network (MP CNN) with both object localization and part selection models without any annotations.

Several works have been done in classifying animal images and each of them has attained a different level of accuracy on their dataset. In one of a unique attempt to classify animals, Yu *et al.* [22] has manually cropped the animal in the image and selected those images that contain the whole image of an animal for classification. With this, they were able to achieve an accuracy of about 82% in classifying 18 species of animals. In contradiction, Chen *et al.* [23] utilized a convolutional Neural Network (CNN) to automate the process of segmentation and identification. They were able to achieve accuracy of about 38.3% in classifying 20 classes of animals. Norouzzadeh [24] used deep neural networks on the Snapshot Serengeti dataset to train the images. They achieved an accuracy of about 92% in classifying 48 species of animals from Tanzania's Serengeti National Park. The accuracy rate for the same dataset was further improved by Villa *et al.* [25] also utilized the very deep convolutional neural network for classifying 20 different animal species from Snapshot Serengeti dataset. Using residual network (ResNet) topology, they achieved an accuracy of about 88.9% in the Top-1 category and about 98.1% in the Top-5 category. In identifying animals from Camera-trap images, many [24], [26] has achieved a decent accuracy. But, their models were based on manually designed feature extraction and moreover, they worked only with few thousands of images. Though feature extraction is one of the preliminary steps in image processing, their use in automated image classification is limited and hence the works suffered from low accuracy rates.

Various dog breed classification model has been developed. In [27], CNN has been used for dog breed categorization. Two popular architectures namely LeNet and GoogLeNet are utilized for classification. The model has achieved an accuracy of 95% with LeNet and 89% with GoogleNet. In [28], different dog breeds are classified using the landmark-based shape representation of the animals. Grassmann manifold is utilized to project the shape of the dog as points. The model has worked on 133 breeds of dogs with 8,351 images and has achieved a 96.5% recognition rate. Similarly, in [29], an appearance model utilizing exemplar-based geometric is used along with face parts to categorize different dog breeds. Accurate localization can improve the performance and the model has achieved a 67% recognition rate in the test data. The model has trained on a dog breeds dataset of 133 classes with 8,351 images. Tensorflow [30], a neural network-based open-source machine learning tool has been developed for several interesting applications like Text recognition [31], and traffic flow prediction [32]. From the above literature, we found the following drawbacks in existing systems;

(i) By far, most of the animal classification system has focused on classifying various animals on a generic level [22]–[26]. The intra-class variance of animals is hardly dealt with.

(ii) Most of the animal image classification system was carried out on very little dataset [33]–[36]. Also, the system has suffered poor accuracy rate, as low as 38% [23] and the highest accuracy achieved so far is 92% [25]

(iii) A few of the models followed the manual approaches for classification and was not completely automated. For instance, manual cropping of images [22] for segmenting the region of interest.

(iv) The training images were biased [22]–[29] and this leads to a higher recall rate for the category with the highest number of images. The effects of a balanced dataset are not studied [4]–[19].

(v) Few works used some form of Deep Neural Network architecture like Alexnet [7]–[9], Residual Network (ResNet) [25], Visual Geometry Group (VGG [11]–[14], Network In Network (NiN) [20]–[22]. The highest accuracy obtained so far was 96.8% with VGG architecture. This leaves room for improvement. Inception architecture is not used so far [6]–[9], [23]–[26], [28].

(vi) Most of the fine-grained classification problems that used part based models relied either on bounding box for localizing objects or annotations for choosing the discriminating parts [16]–[18].

(vii) In some cases, objects were not localized before selecting the discriminative parts [37]. The huge background and its overlap with the foreground object lead to poor classification.

(viii) Most of the works relied purely on Deep Neural Network for the classification purpose [23]–[25], [27]–[29]. The effects of DNN in combination with other techniques like semi-supervised learning were not studied.

(ix) To the best of our knowledge, no previous works have been done in handling the misclassifications arising out of the classification modes. Misclassifications arise out of various reasons and no literature is reported by far in resolving them [13], [19], [21].

The main motivation behind the work is to propose a model that helps mitigate man-animal conflict by developing an animal monitoring and detection prototype that effectively monitors the animals in the wild and detects its presence when it enters the village or crop fields. The animal detected as wild should be correctly identified. This work focuses on classifying the animal on both generic and fine-grained level. The main motivation for this research work is elaborated as follows:

(i) Misclassifications are hardly dealt with [22]. We were interested in rectifying the misclassifications that occur due to various reasons and thereby increase the accuracy of our model.

(ii) The existing models mostly followed a supervised approach for classification. This encouraged us to develop a semi-supervised learning model for automatically classifying unlabeled images too. The adaptability of the classifier to classify unlabeled images is not studied [20], [21], [29], [37].

(iii) As most of the works had a biased dataset [22]–[27], [29], we wanted to know how unbiased dataset affects the performance of the model. So, we have experimented with both biased and unbiased dataset.

With all the above points in mind, we concluded on utilizing semi-supervised learning based Multi-part Convolutional Neural Network for our classification application. In summary, the paper contributes to the following,

(i) Developing Multi-part Convolutional Neural Network architecture with a hybrid feature extraction technique of the Fisher Vector and Stacked Autoencoder for classifying the breeds of animals.

(ii) Minimizing the misclassification from test dataset with Modified Hellinger Kernel Classifier

(iii) Semi-supervised learning approach for handling new classes of true images from real-world scenarios where the ground truth is not available.

(iv) Discusses the effects of balanced vs. unbalanced dataset and the best performance metrics that could be used for both.

The remaining of the paper is organized in the following way: Section 2 explains the preliminary concepts and Section 3 deals with an application scenario for our proposed model. Section 4 describes the proposed methodology. Section 5 discusses the experimental framework and performance metrics. Section 6 details the results obtained and a comparison with the existing system is given in Section 7. The conclusion and future scope are presented in Section 8.

II. PRELIMINARY CONCEPTS

In this section, we brief the feature extraction techniques Fisher Vector and Stacked Autoencoder. Feature extraction techniques can be broadly categorized into two ways. In the first case, the features are directly extracted from the original high-dimensional input. In this case, the extracted features become a subset of the original input. Let the original dataset be denoted by O having N features and let the subset of the original dataset (extracted dataset) be denoted by E and it has n features. This can be represented as:

$$O:\{o1, o2, \ldots, oN\} \rightarrow E:\{e1, e2, \ldots, eN\} \tag{1}$$

$$e_i \in N, \quad i = 1, 2, \ldots, n; n < N \tag{2}$$

In the second case, the features from the original dataset are projected from a high-dimensional space to low-dimensional space. In this case, we say it is as generated dataset, which is nothing but the mapping of the original dataset. Let the original dataset be denoted by O having N features and let the generated dataset be denoted by E and it has M features. This can be represented as:

$$O:\{o_1, o_2, \ldots, o_N\} \rightarrow E:\{e_1, e_2, \ldots, e_M\} \tag{3}$$

$$(e_1, e_2, \ldots, e_M) = f(o_1, o_2, \ldots, o_N) \tag{4}$$

Among the two categories, the Fisher vector belongs to the first type and Stacked Autoencoder belongs to the second type. We propose a Hybrid feature extraction framework that combines Fisher's vector with Stacked Autoencoder. This is done to fully obtain the advantage of both the techniques.

A. FISHER VECTOR (FV)

Let us consider a sample image I for which we generate the Fisher vector. We use Probability density function p with μ parameters. The sample I can be defined by the gradient vector as [38]:

$$G_\mu^I = \nabla_\mu \log p\,(I \mid \mu) \tag{5}$$

The log gradient depicts the involvement of the parameter μ in the vector generation process. The dimensionality of the fisher vector is based on the parameter μ. A plain kernel for the above gradient would be as:

$$k\,(I, J) = G_\mu^{\acute{I}} F_\mu^{-1} G_\mu^J \tag{6}$$

In the above expression, F_μ is the Fisher matrix of the probability density function p.

$$F_\mu = E_{I\sim p}\left[\nabla_\mu \log p\,(I \mid \mu)\, \nabla_\mu \log p(I \acute{\mid} \mu)\right] \tag{7}$$

In the expression above, we can note that F_μ is positive-definite and symmetric in nature, hence it has Cholesky decomposition i.e. $F_\mu = L'_\mu L_\mu$ and K (I, J) can be rewritten as:

$$G_\mu^I = L_\mu G_\mu \tag{8}$$

It is a product between the normalized vectors G_μ. The fisher vector of the sample image I will be denoted by G_μ^I.

B. STACKED AUTOENCODER (SAE)

Though the Fisher vector is one of the state-of-the-art feature extraction techniques, it does suffer from non-linear information loss. To avoid this loss of information, we use Stacked Autoencoder (SAE). Like Convolutional Autoencoder, stacked Autoencoder is also a feed-forward neural network. All the idea is similar to the convolutional neural network, except that we have a stack of layers in SAE. The following defines SAE mathematically [39].

In SAE, each layer in the stack has a separate Autoencoder. Let us consider a single layer in SAE. For a given input X, the corresponding input vector is $\mathrm{x} \in \mathbb{R}^{\mathrm{n}}$ and the activation neuron, a_i for i = 1,2, . . .,m is calculated by $\mathrm{a\,(x)} = \mathrm{f(W_1x + b_1)}$ where $\mathrm{a(x)} \in \mathbb{R}^{\mathrm{m}}$ is the pattern followed by neuron activation, $\mathrm{W_1} \in \mathbb{R}^{\mathrm{mxn}}$ is the weight matrix, and $\mathrm{b_1} \in \mathbb{R}^{\mathrm{m}}$ is the bias. For a non-linear mapping of the dataset from high-dimensional space to low-dimensional space, we use the sigmoid function. After all the pre-training, the output of the neural network is:

$$\hat{x} = f\,(W_2 a\,(x) + b_2) \tag{9}$$

In the above expression, $\hat{x} \in \mathbb{R}^{\mathrm{n}}$ is again the pattern of output, $\mathrm{W_2} \in \mathbb{R}^{\mathrm{mxn}}$ is the weight matrix and $\mathrm{b_2} \in \mathbb{R}^{\mathrm{m}}$ is the bias. For a given input vector $x^{(i)}$, $i = 1, 2, \ldots, p$, we calculate their weight matrices by the back-propagation technique.

$$e^x = \sum\nolimits_{i=1}^{p} \left(\left\| x^{(i)} - \hat{x}^{(i)} \right\|\right)^2 \tag{10}$$

The above expression is the Gradient descent method and is used to reduce the reconstruction error

III. APPLICATION SCENARIO

In this section, we discuss the consequences of man-animal conflict and how could it be mitigated. The proposed classification model could be effectively applied in this scenario.

Man-animal conflict is one of the main threats to the continued survival of animal species and has also impacted the lives of humans. They normally result in a negative impact on human or the animal or both. Such a situation is avoidable in most cases. Man-animal conflict occurs either when an animal enters a human habitation or when a human enters a wildlife zone. Both scenarios can be averted if a monitoring and detection system is employed. It is easier to control the first scenario (animal entering a human habitation) than the second one. In this case, we can utilize a monitoring device to continuously monitor the movement of animals. Fig. 2 demonstrates the working of animal monitoring and detection system.

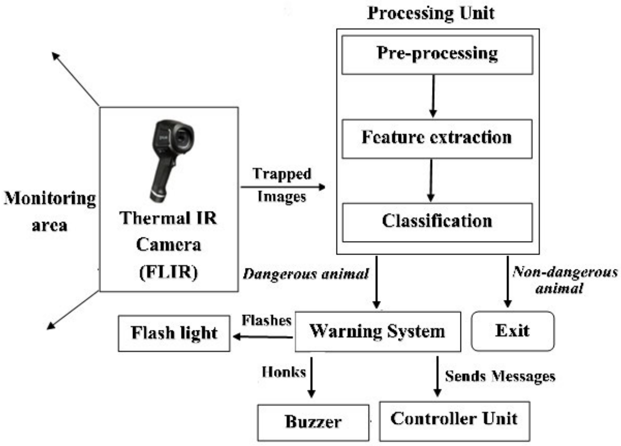


FIGURE 2. Animal monitoring system.

The trapped images are processed and classified with the animal classification model and when the trapped animal is detected as wild, an alert is made. The monitoring and alert system are out of the scope of this paper. The proposed classification and detection model can be employed in scenarios similar to this. This model can be highly useful in detecting animals with high accuracy. One important factor to consider is the capability of the model to differentiate between animals with high intra-class variance. For instance, the model should differentiate between a black cat and a black panther, given in Fig. 3.

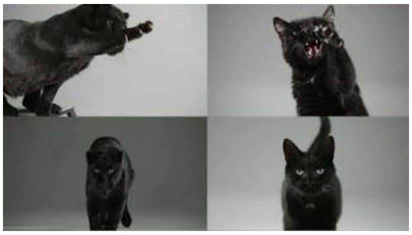


FIGURE 3. Different classes of animals with high intra-class variance: Black panther vs. black cat.

A black panther is more of a wild animal and the system should correctly identify it like a wild animal and alert accordingly. At the same time, alerting for a black cat would make no sense. Accordingly, the main task is to recognize the animal and classify it accurately. For this purpose, we have proposed a coalesced approach of Semi-supervised learning based Multi-part Convolutional Neural Network built on Tensorflow, which can classify animals on a fine-grained level. Our future work would focus on developing a complete animal monitoring and detection system for mitigating the man-animal conflict and would be based on thermal images of animals.

IV. PROPOSED METHODOLOGY

In this section, we discuss the two stages of our proposed model and how they fit together for animal breed classification. The first stage is the generic level classification where the model is trained and tested with Tensorflow. The classification results are analyzed with TensorBoard. Those classes of animals with higher misclassifications are re-trained with Pseudo-labels (in case of unlabeled classes of images) and MHKC (in case of labeled classes of images). The second stage is fine-grained classification where we utilize the Multi-Part Convolutional Neural Network (MP-CNN). The flow of our proposed framework is depicted in Fig. 4.

The features for the MP-CNN are extracted on a two-fold basis. Handcrafted features are extracted from Fisher Vector based Stacked Autoencoder and the deep features are extracted from the multi-part CNN.

A. STAGE 1: GENERIC LEVEL CLASSIFICATION OF ANIMAL SPECIES

In the first step, we embrace the concept of transfer learning for classifying the different animal species. The misclassifications are then re-trained with MHKC and semi-supervised pseudo-labels to improve the classification rate.

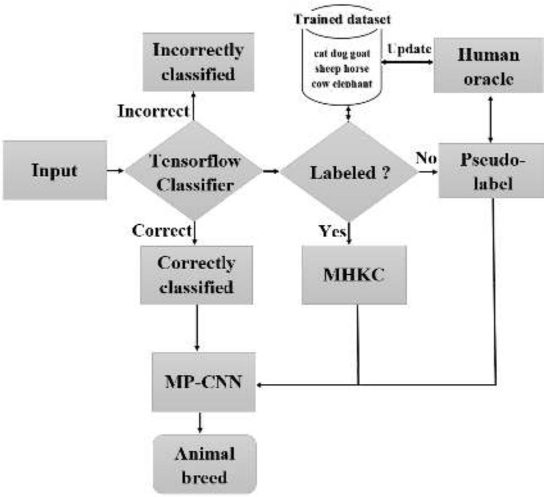


FIGURE 4. The flow of the proposed framework.

1) TRANSFER LEARNING WITH TENSORFLOW

The Tensorflow version of Inception V3 is used for the classification. Originally, Inception V3 by Google is trained on the ImageNet dataset of 1000 classes, which is approximately over 1 million of training images. However, the Tensorflow version has 1001 classes and the 1 extra class is the background class which was not in the ImageNet dataset. The training images are of 224 × 224 dimensional high-resolution color images. To reduce the dimension of an input image, we use a 1 × 1 Conv layer. Fig.5 describes the modified final Inception module. Each of the max pool layers is 3 × 3 with a stride value of 1 and the same padding. At the end of each inception module, we perform channel concatenation. The learning rate was set to 0.005 with a batch size of 100, and a training step of 500.

For each of the test images, the model produces a list of labels along with confidence. Classification is done purely by comparing the bottleneck value of the test image along with all the bottlenecks generated.

2) RE-TRAINING THE MISCLASSIFIED LABELED IMAGES WITH MODIFIED HELLINGER KERNEL CLASSIFIER

Here we discuss how Modified Hellinger Kernel Classifier (MHKC) could be used for improving the classification rate. The output from Tensorflow was decent enough for a typical classification problem. But the results were not satisfactory as there were few misclassifications when it comes to the fine-grained classification of animals. The results from Tensorflow are analyzed using TensorBoard and those classes having the highest misclassifications are fed to MHKC. The training dataset of the misclassified classes is taken along with some background images (non-animal images). MHKC will then process this dataset and will output only the best images for the class considered. In short, it acts as the best image retriever. The number of images obtained as output depends on the value of the ‘‘rank’’ parameter. The rank

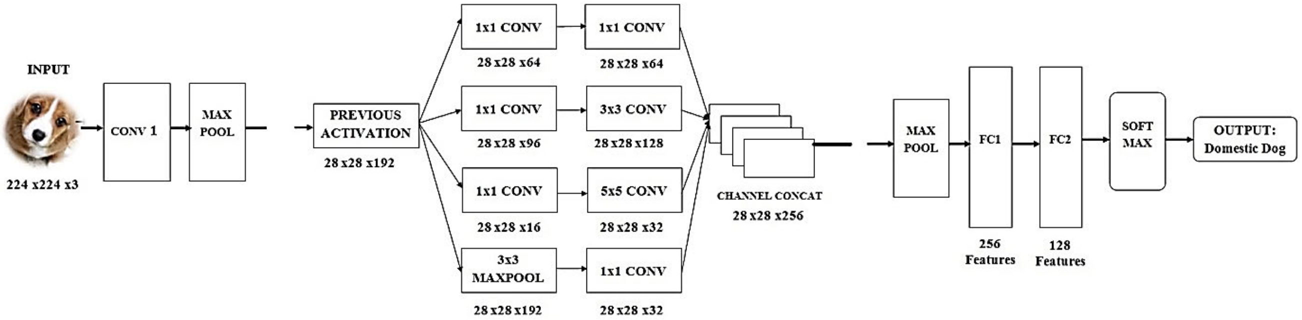


FIGURE 5. Modified Inception v3 module.

value of 100 will produce the best 100 images for the given class. This way, we pre-process the training dataset of the misclassified classes. It is to be noted that the input to MHKC was augmented well enough so that we could retrieve as many as required. The results with MHKC trained dataset were quite better than before. The intuition behind MHKC is that a well-correlated train dataset has better performance in the test data than a class that has some randomly chosen training data. It retrieves the most suitable and well-correlated images for the training data.

3) RE-TRAINING THE MISCLASSIFIED UNLABELED IMAGES WITH SEMI-SUPERVISED LEARNING BASED PSEUDO-LABELS

Although MHKC is one of the best methods for handling misclassifications in trained data, it is not the best approach for unlabeled images. To handle the misclassifications due to unlabeled data, we utilize a different technique called ''Pseudo labeling based on Semi-supervised learning''. In a real-time animal detection system, one cannot expect to encounter only those animals that are trained by the model. The adaptability of the model is a significant factor for any real-time application. Real-time models are dynamic in nature and for such cases, Semi-supervised learning is a better choice when compared to supervised learning. Indeed, both human and animal learning models are highly unsupervised. Hence, we have imparted the semi-supervised learning approach in our model, so that it can train itself for any new classes of animals it encounters. Pseudo- labeling is one of the most efficient yet simple methods for performing the semi-supervised approach. The working of pseudo-label is as follows;

(i) Initially, we train the network in a supervised manner, i.e., using the labeled train and test data. The hyper-parameters of the network are adjusted to achieve good results.
(ii) On the same network, we now train our unlabeled datasets and try to label them with a pseudo-label or quasi label.
(iii) The newly generated quasi label is concatenated with the original training label.
(iv) Similarly, the features of the quasi-labeled dataset are concatenated with the features of the training dataset.
(v) Finally, we again train the network with the new set of labels (from iii) and features (from iv).

When an unlabeled true image arrives, the model would try to classify it to one of the closely related pre-defined labels. However, the classification scores or accuracy will be very low, indicating that the image is misclassified. Images with lower classification score are assumed to be unlabeled data and will be back-propagated to the network for training. The basic assumption of classification is clustering, where the labeled data of individual classes are clustered. On this basis, the unlabeled data are given pseudo-labels. The pseudo-labels are generated by the clusters with which it has a higher feature affinity i.e., considering the complex feature relationship among the labeled and unlabeled data in the cluster set. Thus, for each new unlabeled class of images, a new pseudo-label is generated making them a separate class. We use the trained network to generate pseudo labels for unlabeled data. We simultaneously train both the trained and pseudo labels for each of the new interactions and the weights are adjusted accordingly. The overall process involved in classifying a true unlabeled image via pseudo labeling is depicted in Fig. 6.

The set of true images (unlabeled images or images without ground truth) are fed to the pre-trained neural network and the features are extracted from the unlabeled images. These images are then clustered based on the extracted features. Cluster defined pseudo-labels are generated for the unlabeled images. To confirm the pseudo-label, we search for a supporting sample (with a confidence score) in the training dataset. If a sample is found, then the pseudo label is confirmed and the classifier outputs the label. Instead, when a supporting sample is not found, we solicit a human annotator to resolve the pseudo-label. The resolved label is finally updated to the training set. Next time, when a similar instance of image arrives, we will have a supporting sample and hence the human annotator will not be required.

B. STAGE 2: FINE-GRAINED ANIMAL BREED CLASSIFICATION OF ANIMAL SPECIES WITH MP-CNN

In this step, we classify the various breeds of animals using a multi-part based CNN model. For this, we propose a hybrid feature extraction framework that combines

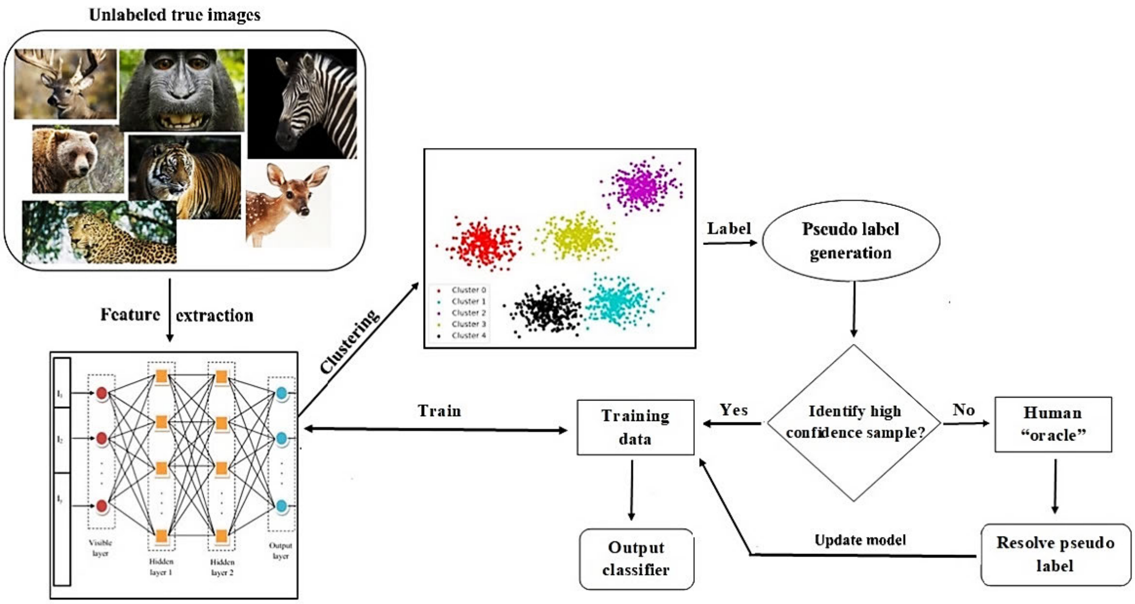


FIGURE 6. Semi-supervised learning based pseudo-labeling.

hand-crafted global features with that of deep features extracted from CNN. The global features are extracted from a hybrid technique named Fisher Vector based Stacked Autoencoder. The deep features extracted from the CNN will be taken as local features. The hybrid feature extraction framework provides a very rich set of features that are required for the fine-grained classification.

1) UNSUPERVISED FEATURE LEARNING VIA FISHER VECTOR BASED STACKED AUTOENCODER

With the advancement in deep learning, one can easily extract the activations of a pre-trained neural network model. But, when these features are used as Global features, the outcome may not be optimal [40]. Fine-tuning a CNN may not improve the performance in all cases. To leverage the power of CNN, we use the activations of CNN as local features instead of global. We introduce a novel feature extraction framework based on Fisher Vector (FV) and Stacked Autoencoder and we name it as Fisher vector based Stacked Autoencoder (FVSAE). The brief description of FV and SAE is discussed in Section II. The algorithm for FVSAE is given below [41]:

Algorithm: Fisher Vector Based Stacked Autoencoder for Feature Extraction

Input: Feature sample for training x $\in \mathbb{R}^n$, Labels for training y $\in \mathbb{R}^n$, an odd number of hidden layers for SAE h_1, extracted features from SAE e_1 and features from the Fisher vector f_1.

Output: Overall Features Extracted f_2

(i) Pre-train the hidden layers based on the sample feature x with a constraint that the bottleneck layer to contain only e_1 neurons

(ii) Tune the network based on the training label y with stochastic gradient descent and backpropagation techniques

(iii) Extract the node values of e_1 from the bottleneck layer and try reconstructing them on a new dataset

(iv) Compute the fisher vector G_{μ}^{i}, $i = 1, 2, \ldots, m$ for the i^{th} features in the new dataset

(v) Re-arrange the feature in descending order based on fisher vector value G_{μ}^{i} and select the first f_2 features

We apply the features extracted from FVSAE to the convolutional layers.

2) MULTI-PART CONVOLUTIONAL NEURAL NETWORK

We built a Multi-Part Convolutional Neural Network (MP-CNN) model for the animal breed classification. For the sake of computation cost, we cropped our 224 × 224 dimensional images into 32 × 32. The first part corresponds to the multi-part based model where we extract highly discriminative viewpoint invariant features from the parts of the animals. The second part corresponds to the hand-crafted Fisher vector based feature extraction where we extract local features for the model. The features vector obtained from this hybrid technique is of high dimension and therefore we use Principal Component Analysis (PCA) to reduce the dimensions of the features. The features from both are combined in the fully connected layer FC2. Fig. 7 depicts the proposed framework model of MP-CNN.

MP-CNN can be viewed as a two-level model namely object localization and part selection, as illustrated in Fig. 7. Both these levels do not include any kind of annotations or bounding boxes. The object level model localizes the object through patch selection, for which we utilize the

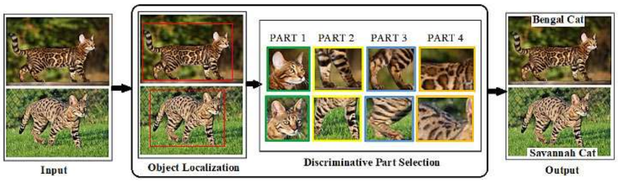


FIGURE 7. Overview of proposed MP-CNN model. The object level model localizes the animal and the discriminative parts are selected in part level model.

pre-trained FilterNet [42]. The primary motive of object localization is to eliminate the larger background and also to avoid the object overlap with the background. The selected candidate patches are aligned to get the localized image of the object (in our case, the object is the animals). Following the FilterNet, we feed-forward the selected patches to DomainNet [43], for extracting features specific to the generic category of animals. (For instance, it will extract features relevant to the dog rather than pug or poodle).

The part level model chooses the discriminative parts for fine-grained classification. The part level model also considers the two sets of spatial relationships. The first set is between the object and the parts and the second is the relationship among the parts. Based on part clustering, the discriminative parts are chosen and the features are extracted. The object and part level feature together aids in classifying the animals on a fine-grained level.

Level 1: Object localization model

CNN requires a large number of training data to achieve a significant results. Instead of going for any random data augmentation, we utilize the bottom up process for data expansion. Bottom up approach combines pixels into regions and can generate several thousands of image patches where we can find objects. We utilize the widely used selective search [44] algorithm for choosing the candidate image patches, which provides a multi-view and multi-scale patches for the original image. This type of data augmentation is more relevant to the training image and can be used for effective training of CNN and a higher classification accuracy can be achieved. However, all the patches cannot be taken for training, as they contain noises to a certain extent. We choose only the relevant patches and remove those unwanted patches with FilterNet, which is a pre-trained CNN. The FilterNet is trained on the ImageNet dataset and we have fine-tuned it on our training data. The selection confidence score for the activation neuron in the Softmax layer is set to the subcategory of the input image. We finally set a threshold value to decide whether to select the candidate patch or not. Fig. 8 illustrates the candidate patch selection model.

The patches selected from FilterNet are warped properly and trained on another CNN called DomainNet. From this, we extract features specific to the primary categories of the animals. It is to be noted that many useful patches can be obtained from a single image and this is an efficient data augmentation technique where one could extract a lot of most meaningful features. Furthermore, DomainNet by itself is a fine-grained classifier since it is built on patches extracted from FilterNet. The features extracted can aid us in building an efficient part detector.

The selected patches can be used in the testing phase to get the label of the image. This is achieved by feed forwarding the DomainNet with the patches selected by FilterNet. For all the given patches, we then calculate the classification distribution in the Softmax layer and get a prediction by averaging the Softmax distribution. One important hyper-parameter here is the confidence threshold. This decides upon the quality and quantity of the selected patches and we set it to 0.9 to achieve a reasonable training time and the best validation accuracy.

Level 2: Part selection model

For a fine-grained classification, identifying discriminative parts is essential. Previous works have either used part annotations directly on the input image or annotated the parts on the selected patches. One other common point in most of the literature is that the spatial relationship among the parts and between the parts and the objects is ignored. We propose a part selection approach that neither uses part annotation nor ignores spatial relationships. This approach captures the local and subtle discriminations in the images and aids in fine-grained classification.

The part selection consists of two steps. In the first step, the discriminative parts are chosen through a spatial relationship between the object and the parts, as depicted in Fig. 9. In the second step, we cluster the parts based on their semantic meaning.

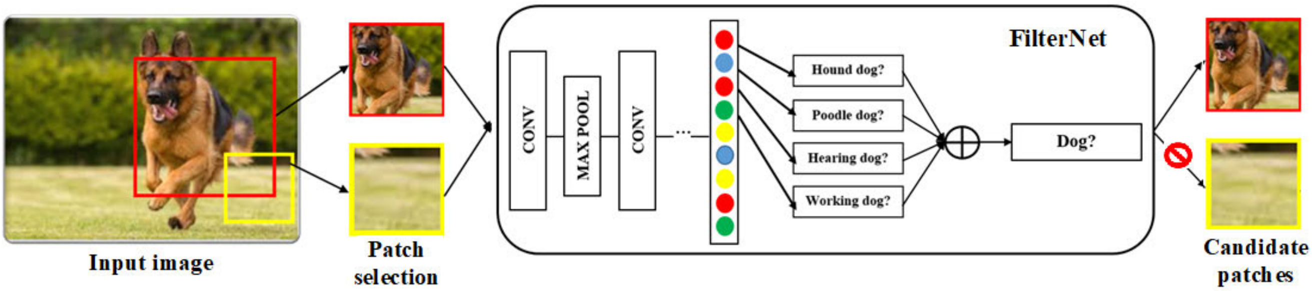


FIGURE 8. Candidate patch selection in object localization model. FilterNet is employed to filter out background patches and choose patches that are relevant to the primary level classification.

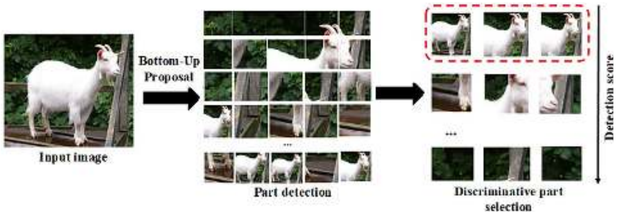


FIGURE 9. **Illustration of the part-level bottom up approach. The discriminative parts are chosen based on the detection score.**

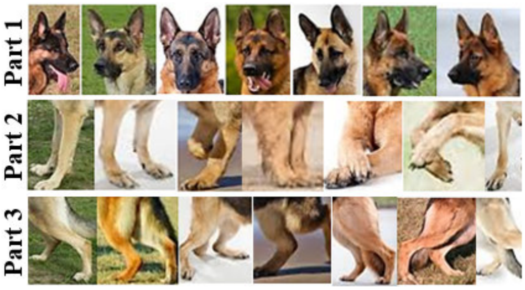


FIGURE 10. **Illustrations of the part selection of our model. Different set of filters represents different parts of the animal. Each row corresponds to different parts. Top: head, middle: front leg, and bottom: hind leg.**

a: OBJECT-PART SPATIAL RELATIONSHIP MODEL

We obtained the object regions from the previous step and on this, we apply the object part spatial relationship model to select the discriminative parts. This model comprises two different relationships namely object spatial relationship and part spatial relationship. The first one is the spatial relationship between the object and the parts and the latter one is the spatial relationship among the different discriminative parts. With object localization model, we can localize the region of the object O_r for a given image I. We define the part selection model, driven by the object-part spatial relationship as follows.

Let us denote the candidate image patches selected by patch selection by P_c and let the parts selected from the candidate patches be denoted by P, where $P = \{p_1, p_2, \ldots, p_n\}$ and n is the number of discriminative parts. The spatial relationship between the object and the parts is achieved through an optimization problem that considers both object spatial $\Delta_{box}(P)$ and part spatial $\Delta_{part}(P)$ relationships. The selected parts should satisfy both these conditions and a score function is defined for these as

$$\Delta(P) = \Delta_{box}(P)\,\Delta_{part}(P) \tag{11}$$

Finally, the object-part spatial relationship is achieved through an optimization function defined by

$$P^* = \arg\max_{P_c} \Delta(P) \tag{12}$$

The score function ensures that only the discriminative and representative pats are selected. The product operation in the score function is chosen based on similar works [35]–[37]. Fig. 10 depicts the sample results for the various parts selected by the model.

Object spatial relationship: The larger background noise and the smaller discriminative parts can be eliminated by considering the spatial relationship between the object and the parts. The discriminative parts lie within the object and we define the object spatial relationship following this intuition.

$$\Delta_{box}(P) = \prod\nolimits_{i=1}^{n} f(O_r(p_i)) \tag{13}$$

The object localization or the region of the object is denoted as $f(O_r(p_i))$. The parts are selected only if it satisfies the below condition.

$$f(O_r(p_i)) = \begin{cases} 1, & IoU(p_i) > t \\ 0, & otherwise \end{cases} \tag{14}$$

The overlap between the object and the parts (p_i) is defined using the Intersection-over-Union (IoU) $IoU(p_i)$. The product operation in the object spatial relationship ensures that all the selected discriminative parts remain inside the object. When the IoU equals 0, then no discriminative parts are selected.

Part spatial relationship: Generally, the selected parts may overlap with each other and this leads to missing some discriminative parts. Hence, we consider the spatial relationship between the parts. The patches from DomainNet gives clear discrimination among the parts. The spatial relationship among parts is given by;

$$\Delta_{part}(P) = \log\left[(A_U - A_I - A_O) + \mu\left(P_{d_{A_U}}\right)\right] \tag{15}$$

In the above equation, A_U is the area of union for the n parts, A_I is the area of intersection for the n parts and A_O is the area outside the region. The mean of the patches from DomainNet is represented by $\mu(P_d)$. The area of union of $\mu(P_d)$ is given by;

$$\mu\left(M_{A_U}\right) = \frac{1}{|A_U|}\sum\nolimits_{i,j} P_{d_{i,j}} \tag{16}$$

$P_{d_{i,j}}r$ efers to the value of the patches at pixel (i, j). The part spatial relationship reduces the overlap between the parts with $log(A_U - A_I - A_O)$. Subtracting the A_I and A_O from A_U ensures that there is very minimal overlap and also the selected discriminative parts have a larger proportion in the object region respectively. The part spatial relationship also tries to have a maximal region for the object and this is achieved with $log\left(\mu\left(P_{d_{A_U}}\right)\right)$. Summing up both these operations will have a net effect on the object-part spatial relationship.

b: PART CLUSTERING

Generally, the selected parts will neither be aligned nor ordered. However, we can cluster the parts based on their semantic meaning. Existing works have either used key points or labels to cluster the parts. In one of the unique techniques for clustering, [21] has clustered based on the neurons of the convolutional layers. Inspired from their approach, we utilize the same approach of finding the cluster pattern among the hidden convolutional layers. Despite the multi-scale and multi-view poses of the animals, we found a pattern among the neurons in the middle layers of the convolutional layer. Each of the parts of the animals was represented by a set of neurons and we performed a spectral clustering on the middle layer neurons. In particular, among the five convolutional layers, the pattern was clearly represented in the third layer. The clusters are chosen by computing the similarity matrix among the two middle layer neurons represented as n_1 and n_2. The similarity matrix denoted by '$S(i, j)$ is a cosine similarity function on the weights of the two middle layers neurons n_1 and n_2. The clusters are segregated through a spectral clustering on the matrix S. We set the cluster value $c = 4$, each for the four discriminative parts namely head, body (the upper part extending from the neck), fore and hind legs. The middle layer neurons are chosen based on standard practices. With the 10% of data for the validation set, we used the grid search to choose the appropriate layers for the cluster pattern analysis. The third layer presented a clear cluster pattern when compared to the second and fourth layers. With this intuition, we chose the discriminative parts and they are trained separately for the final classification. Following the clustering, the parts are aligned. The selected parts are warped to fit the size of the input image. The images are fed to the convolutional layer and are feed forwarded to the next to last convolutional layer and an activation score is produced based on the scoring function. The score function is given for each of the neurons and the score of one complete cluster is estimated as its cluster score. The parts are aligned to the cluster with the highest cluster score.

Table 1 is the activation shape and activation sizes of different layers. Max pooling layers will not have any parameters.

TABLE 1. Description of MP-CNN layers.

Layer	Activation Shape	Activation size	No. of parameters
Input	(32,32,3)	3,072	0
Conv1 (f=6, s=1)	(28,28,6)	4,704	208
Pool 1	(14,14,6)	1,176	0
Conv2 (f=10, s=1)	(10,10,10)	1,000	208
Pool 2	(5,5,10)	250	0
Conv3 (f=16, s=1)	(10,10,16)	1,600	416
Pool 3	(5,5,16)	400	0
Conv4 (f=20, s=1)	(3,1,20)	60	416
Conv5 (f=24, s=1)	(3,1,24)	72	512
FC1	(120,1)	120	48,001
FC2	(84,1)	84	10,081
Softmax	(27,1)	27	

It can also be noticed that Conv layers have fewer parameters and a lot of parameters tend to be in the fully connected layers of the neural network. The activation size tends to go down gradually as you go deeper into the neural network. If it drops too quickly, then the performance will not be good.

ADVANTAGES OF THE PROPOSED SCHEME

The proposed framework has leveraged the power of several state-of-the-art techniques to produce better results. Some of the advantages of our approach are given below.

1) The open-source technology Tensorflow saves a sizeable amount of time from building and training a neural network from scratch.
2) While it is time-consuming to initially train the dataset in Tensorflow, but any amount of new data can be added to the training dataset and training the new ones takes very little time, when compared to direct training in DNN.
3) Transfer learning makes it is quite easy to check with different deep learning architectures.
4) MHKC utilizes the power of SVM with L2 normalization, thereby producing a better result than counter methods [23].
5) The semi-supervised learning approach based on Pseudo-labels aids in developing a full-fledged automated animal classification model.
6) The non-linear information loss is handled by combining the Fisher Vector with Stacked Autoencoder and thereby extracting the highly rich set of local features.
7) The viewpoint invariance is eliminated with MP-CNN.
8) Our coalesced approach has better accuracy and lesser processing time in totality when compared to building and testing only with a DNN.

V. EXPERIMENTAL FRAMEWORK

In this section, we discuss the equipment, dataset, system requirements, and experiments undertaken in detail. We also discuss the performance metrics considered for evaluating our proposed framework model.

A. EQUIPMENT

In this work, we used the Forward Looking InfraRed (FLIR) [45] thermal camera of model e40, for capturing the images of animals during day and night time. We have utilized both thermal and visible images from FLIR. Thermal imagers are one of the perfect tools for night vision applications. They can see through the darkness without the need for light. They work on the principle of heat energy and so they can detect animals, as animals are hot-blooded animals. Moreover, thermographic cameras can ignore camouflage too. The IR resolution of e40 is160 × 120 with a thermal sensitivity of 0.07° and a temperature range of −20 to 650°. FLIR works on Long-Wave Infrared Band (LWIR), which is the most preferred one for detecting animals.

B. DATASET

For the experiments, we have taken the dataset from ImageNet as well as Google images. Also, we have captured a few images (both visible and thermal images) with a FLIR thermal imaging camera. For the test cases, we utilized the visible images captured with the FLIR thermal camera. ImageNet [46] is an image dataset developed for the 2012 ImageNet Large Visual Recognition Challenge (ILSVRC 2012). The dataset is structured according to the WordNet hierarchy, where each of the WordNet is expressed by many word phrases called Synonym set or synset. Each synset has more than 1000 images on average and is human annotated. For our model, we have taken a total of 35992 images (sample is shown in Fig. 11) belonging to 27 different animals of 7 categories (listed in Table 2). These 7 categories of animals can be broadly classified into Livestock, Caprine animals, Domesticated animals, and Work animals. Livestock includes milch cow, Billy goat, Nanny goat, and Domestic sheep. Caprine animals include Mountain sheep, Wild sheep, and Wild goat. Work animals include Stallion, Mare, Mounts, Wild horses, African elephants, and Indian elephants.

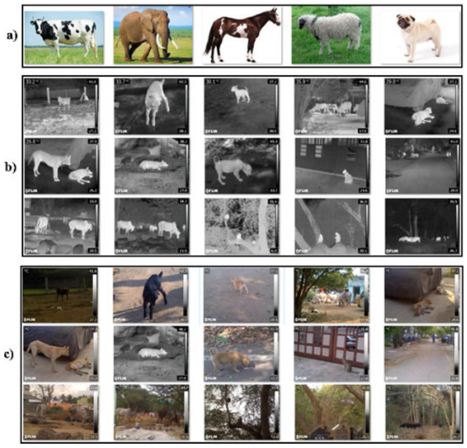


FIGURE 11. Sample images from the dataset a) Images from ImageNet b) thermal images captured with FLIR c) Visible images corresponding to the thermal images in (b), captured with FLIR.

TABLE 2. Dataset description.

Category	Class	Count	Occluded images	Partially visible images	Far away image	Multiple characters per image
Sheep	Domestic sheep	1623	142	434	43	78
	Wild sheep	911	67	564	56	39
	Mountain sheep	1165	266	211	133	38
Horse	Stallion	1485	283	76	101	521
	Mare	1902	334	98	34	86
	Wild horse	1187	787	34	67	23
	Mounts	1249	235	164	45	532
Goat	Billy goat	1114	513	76	78	68
	Nanny goat	1240	348	421	85	301
	Wild goat	1229	532	69	14	46
Elephant	African elephant	2277	569	677	75	21
	Indian elephant	1650	532	897	12	19
Dog	Bird dog	1021	346	89	87	259
	Domestic dog	1603	446	34	43	792
	German Shepherd	1741	78	76	57	830
	Hearing dog	31	1	0	0	8
	Hunting dog	1222	226	76	54	151
	Poodle dog	1345	149	321	34	453
	Hound dog	1249	578	65	87	275
	Pug	1261	480	342	85	211
	Working dog	1340	479	87	32	537
Cow	Milch Cow	1186	325	43	70	214
Cat	Burmese cat	1169	742	21	43	79
	Egyptian cat	977	648	96	76	48
	Persian cat	1662	356	34	89	135
	Pussycat	1414	754	87	12	329
	Siamese cat	1739	146	43	76	265
7	27	35,992	10,362	5135	1588	6358

Our dataset has also included some of the challenging image conditions such as partially visible images, occluded images, too far images and images with different animals. Our original training dataset is unbalanced with a varied number of images for each animal. For instance, the hearing dog had the least amount of images of about 31 and the African Elephant had the highest number of images of about 2277. In addition to the experiments carried out with an unbalanced dataset, we have also experimented with a balanced dataset, where each of the class has 900 images exactly and hearing dog class is omitted.

To have a better understanding of the effects of the training dataset, we had five different categories of the dataset and each category differed in the number of images they hold for train and test dataset. Category I contain 80% of the training dataset and the remaining 20% is taken for testing. Similarly, Category II to V contains 60%, 50%, 40% and 20% of training dataset respectively and their remaining is taken for their respective testing.

C. SYSTEM REQUIREMENTS

The experiments were carried out in single NVIDIA GeForce 940M Version 376.82 GPU based laptop with Intel(R) Core(TM) i5-5200U CPU @ 2.20GHz processor and 8GB RAM. For storage purposes, we used 1 TB Seagate Hard disk drive. Tensorflow was run on Windows based Docker

Environment. MATLAB R2015b was used for running Modified Hellinger Kernel Classifier. Neural Network was run on MATLAB R2018a.

D. EXPERIMENTS

The complete dataset was thoroughly studied with various task and the effects of our proposed scheme on them are noted. Following are the tasks undertaken;

(i) What is the accuracy when the classes have a balanced dataset?
(ii) What is the accuracy when the classes have an unbalanced dataset?
(iii) How will be the performance when classes have challenging images of animals?
(iv) How will be the performance when classes have a complete image of animals?
(v) How will be the performance when classes have mixed images (both challenging and complete images)
(vi) How good is the performance of the proposed model when compared to existing neural network architectures?

In task 1 we consider an equal number of training images for all classes of animals. In our case, we have taken 900 training images for all the classes of animals except Hearing Dog. Task 2 is more of a practical one, as it is not possible to have equal training for all classes. But this is done to understand the effects of a balanced dataset vs. unbalanced dataset. Images containing the complete picture of animals are considered for Task 4, whereas Task 3 takes only the challenging images as training data like Partially visible images, occluded images, far away images. Task 5 includes both the data from Task 3 and Task 4. We evaluate the performance of our proposed model with that of existing architectures in Task 6.

E. PERFORMANCE METRICS

The performance of our proposed model is assessed with a few usual metrics like Accuracy, Specificity, Sensitivity, and Precision. Accuracy is the proportion to which the model is correct or perfect. Specificity is the percentage to which the model is exact. Sensitivity is the ability of the model to recall. Precision is the Positive Predictive Value (PPV) of the model i.e., being accurate. To calculate these four factors, we find the True Positive (TP), True Negative (TN), False Positive (FP) and False Negative (FN) values. With these values, we calculate the accuracy, precision, specificity, and sensitivity of the model. Each of them is given below;

$$Accuracy = \frac{TP + TN}{TP + TN + FP + FN} \quad (17)$$

$$PPV\ or\ Precision = \frac{TP}{TP + FP} \quad (18)$$

$$Sensitivity = TP/(TP + FN) \quad (19)$$

$$Specificity = TN/(TN + FP) \quad (20)$$

For assessing the performance of MHKC, we have used precision at 'rank n' metric. It corresponds to the number of relevant images that are obtained in the first spot. These are the top results for the given dataset. Precision at rank 25 will retrieve the top 25 best images from the given dataset.

VI. RESULTS AND DISCUSSION

This section discusses the results and is evaluated against the performance metrics discussed above.

A. GENERIC CLASSIFICATION

In this section, we discuss the results of generic classification. The experiments were conducted on a dataset of about 35,992 images. To test the effects of different ratios of training and testing dataset, we categorized the dataset into 5 categories each consisting of 80%, 60%, 50%, 40% and 20% training data respectively. Table 3 discusses the performance metrics of generic level classification.

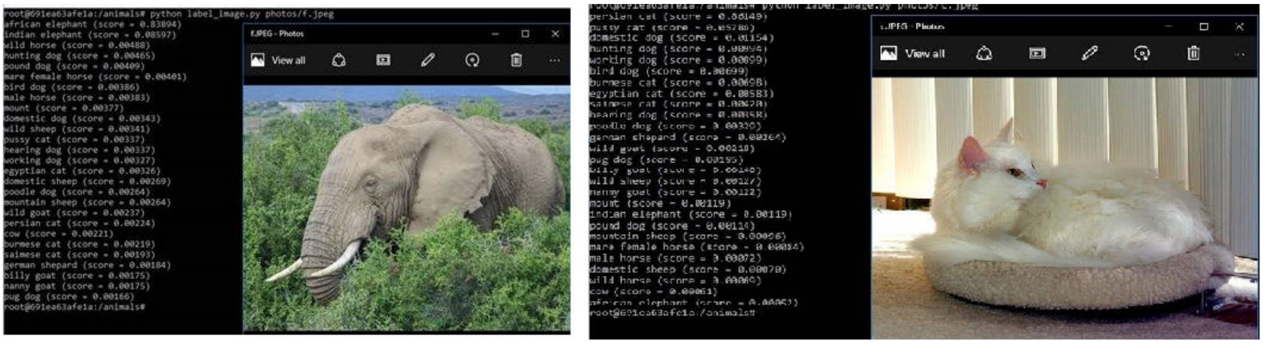
TABLE 3. Performance evaluation of generic classification.

Set	Train data	Test data	Accuracy	Specificity	Precision
I	20%	80%	0.962	0.882	0.952
II	40%	60%	0.968	0.902	0.960
III	50%	50%	0.973	0.922	0.965
IV	60%	40%	0.980	0.942	0.975
V	80%	20%	0.988	0.962	0.985

We also noted the influence of the number of images and their role in testing accuracy. It can be noted from Table 3 that, the accuracy is higher for those classes with a high number of training images. The more the training data, the more the model can learn. Other factors such as Specificity and Precision also increased with an increase in training data. In our experiment, the African elephant constituted the highest number of images of about 2277 and it had the highest testing accuracy in all 5 categories of the dataset, whereas the Hearing dog had the lowest accuracy in all categories. Not all animals were perfectly classified. Fig. 12 represents a sample for Positive and Negative classification.

When it comes to differentiating the breed of animal, we found few misclassifications. In specific, few of the African elephants were misclassified as Indian elephant, Billy goats were misclassified as nanny goats, and Burmese cats were misclassified as Pussycats and Hunting dogs were misclassified as Bird dogs. All these misclassifications were mainly due to the similarities among the animals. In totality, it is found that training the dataset is the crucial part of Tensorflow. Care has to be taken in choosing the appropriate images for each of the classes to be classified. The number of images considered also affects the final accuracy.

The effects of balanced and unbalanced dataset are represented in Fig. 13. From the figure, it is clearly evident that an unbalanced dataset has some effect on the accuracy. For the 80:20 ratio of the training-testing dataset, a balanced dataset has around 93% accuracy, whereas an unbalanced



```
root@691ea63afe1a:/animals# python label_image.py photos/f.jpeg
african elephant (score = 0.83894)
indian elephant (score = 0.08597)
wild horse (score = 0.00488)
hunting dog (score = 0.00465)
pound dog (score = 0.00409)
mare female horse (score = 0.00401)
bird dog (score = 0.00386)
male horse (score = 0.00383)
mount (score = 0.00377)
domestic dog (score = 0.00343)
wild sheep (score = 0.00341)
pussy cat (score = 0.00337)
hearing dog (score = 0.00337)
working dog (score = 0.00327)
egyptian cat (score = 0.00326)
domestic sheep (score = 0.00269)
poodle dog (score = 0.00264)
mountain sheep (score = 0.00264)
wild goat (score = 0.00237)
persian cat (score = 0.00224)
cow (score = 0.00221)
burmese cat (score = 0.00219)
saimese cat (score = 0.00193)
german shepard (score = 0.00184)
billy goat (score = 0.00175)
nanny goat (score = 0.00175)
pug dog (score = 0.00166)
root@691ea63afe1a:/animals#
```

```
root@691ea63afe1a:/animals# python label_image.py photos/c.jpeg
persian cat (score = 0.86149)
pussy cat (score = 0.05786)
domestic dog (score = 0.01154)
hunting dog (score = 0.00994)
working dog (score = 0.00899)
bird dog (score = 0.00699)
burmese cat (score = 0.00698)
egyptian cat (score = 0.00583)
saimese cat (score = 0.00478)
hearing dog (score = 0.00358)
poodle dog (score = 0.00320)
german shepard (score = 0.00264)
wild goat (score = 0.00218)
pug dog (score = 0.00195)
billy goat (score = 0.00140)
wild sheep (score = 0.00127)
nanny goat (score = 0.00122)
mount (score = 0.00119)
indian elephant (score = 0.00119)
pound dog (score = 0.00114)
mountain sheep (score = 0.00096)
mare female horse (score = 0.00084)
male horse (score = 0.00072)
domestic sheep (score = 0.00070)
wild horse (score = 0.00069)
cow (score = 0.00061)
african elephant (score = 0.00057)
root@691ea63afe1a:/animals#
```

FIGURE 12. (a) Positive sample – African elephant classified correctly with 83% accuracy and (b) negative sample -Pussycat misclassified as a Persian cat with 86% accuracy.

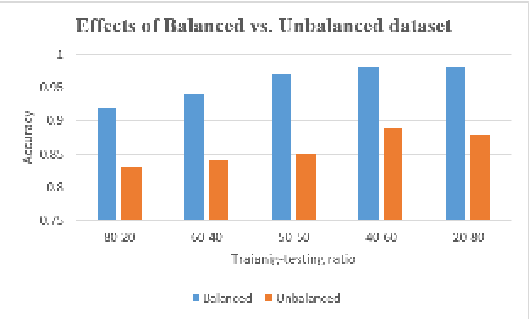


FIGURE 13. Accuracy of balanced vs. unbalanced class.

dataset has only 82% accuracy. The difference between the two categories is hard to neglect. Generally, it is good to have a balanced dataset for any classification problem. An imbalanced dataset tends to have a higher recall rate for the class with the highest training data. The class with the lowest training data will then have higher chances of misclassification. In our dataset, hearing dogs constituted minimal training data and it had the highest misclassification in the test data, whereas Elephants scored high for having higher training data. This imbalanced dataset concept is more complicated in a multi-class classification problem, where more than one class may have minimal data.

The imbalance data problem can be eliminated by balancing the dataset in a way that, all the classes have almost an equal number of train data. However, it is not always the case that a balanced dataset performs well. If the classes are balanced, then we will miss some valuable patterns in the dataset. In such cases, it is better safe to have a large dataset through unbalanced. Balancing a large dataset by augmentation techniques doesn't make much difference in the overall results. There exists a trade-off in every application. The evaluation metric differs for both balanced and imbalanced dataset. The balanced dataset can be evaluated for accuracy, where an imbalanced dataset should focus much on Precision and recall. We have considered all three measures for evaluating the performance of our model. The model has higher performance when training images have only a complete picture of the animal. In contrast, the performance of challenging images class is very low. A combination of both classes has a decent performance.

B. HANDLING MISCLASSIFICATIONS

In a real-world scenario, when a true image arrives, the model will classify that as one of its closest matching animals. Generally, misclassifications could happen in two cases. In the first case, an animal that is trained by the model is misclassified. In the second case, a new class of animal that has not been trained is misclassified. For clarity, we illustrate the two cases separately.

1) CASE 1: MISCLASSIFICATION IN THE TEST DATASET

The output from the first-level classification was quite good except for a few classes. It is insightful to analyze the misclassified classes from the test dataset. We employed TensorBoard to distinguish the misclassifications from correct classifications.

TensorBoard lists the misclassified classes in descending order. Fig. 14 depicts the top 7 misclassified classes. Hearing dogs constituted the highest misclassification rate. We retrained the training dataset of misclassified images with MHKC and the training process can be quantitatively accessed by ranking the training images. Fig. 15-a) represents the training accuracy of horse class. The highest score for the chosen images is 6.60 and the least one is 1.96.

Fig. 15-b) represents the precision-recall curve for the class horse training set. The precision for the training dataset is found to be 100%, which is a good indicator for better testing accuracy. We have improved the accuracy of misclassified horse images to 100% with MHKC.

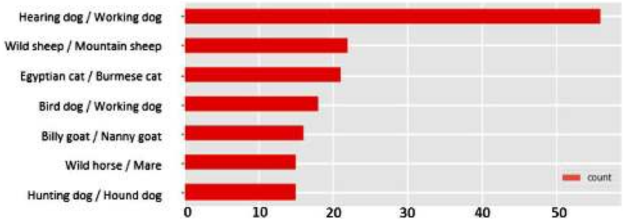


FIGURE 14. TensorBoard listing classes with highest misclassification.

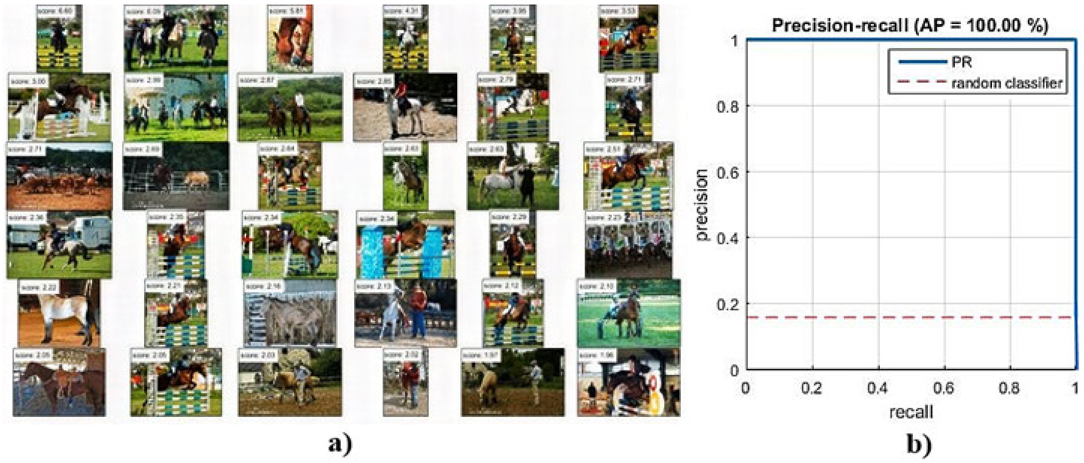


FIGURE 15. a) Training accuracy of horse class with MHKC b) Precision-recall curve.

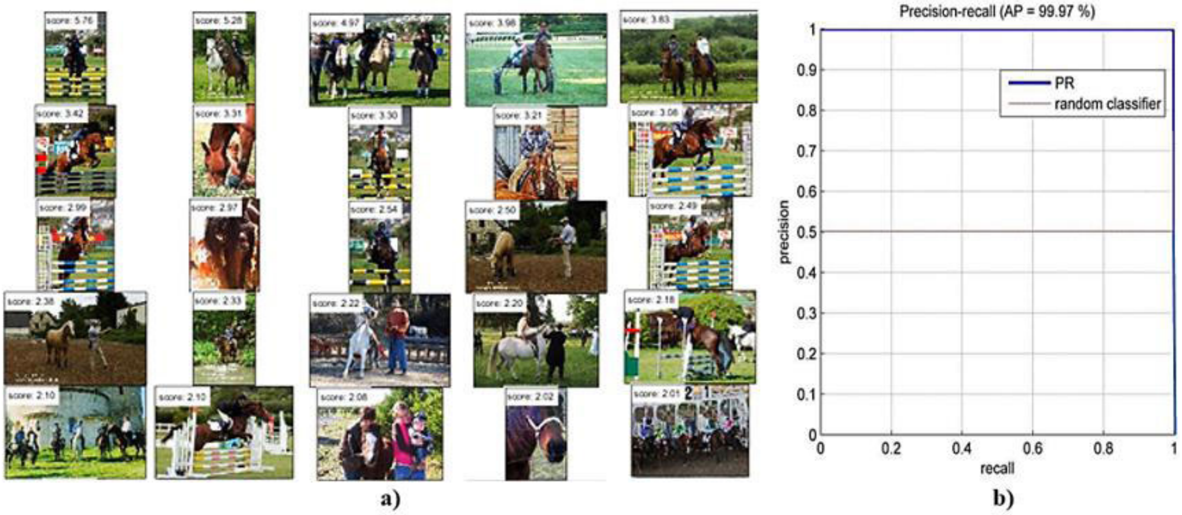


FIGURE 16. a) Testing accuracy of horse class with MHKC and b) precision-recall curve.

With 100% PR on training data, we achieved 99.97% on testing data. This is far better than the accuracy of Tensorflow and the same is depicted in Fig. 16-a) and 16-b) respectively. The feature vector is pre-computed and so the accuracy of the classifier corresponds merely to its kernel functions.

2) CASE 2: MISCLASSIFICATION DUE TO A NEW CLASS OF IMAGE (REAL-WORLD DATA)

To compare the performance of pseudo labeling with that of MHKC, we fed an unlabeled image to both MHKC and Pseudo labeling techniques and the results were obtained.

Fig. 17 depicts the results of an unlabeled image pig, via the two techniques MHKC and Pseudo- labeling. It could be observed that MHKC is comparatively complex for an unlabeled image than its counter technique. Furthermore, MHKC requires manual training for the unlabeled input which is computationally expensive and is also not appropriate for the real-time systems. Conversely, Pseudo- labeling can accurately classify the new class of animal with a generic label, which is later resolved by a human oracle. Though MHKC is a cost-effective technique for handling misclassifications in labeled images, it is not the best choice for unlabeled images. Since Pseudo-labeling belongs to the semi-supervised learning technique, it satisfies the requirements of a real-time animal detection and classification system. Pseudo-labeling is most effective when used with Generative Adversarial Networks instead of CNNs. But, building such a model is beyond the scope of this work.

C. FINE-GRAINED CLASSIFICATION WITH MP-CNN

The ultimate goal of this research is achieved with MP-CNN. When compared to Tensorflow, MP-CNN had a better performance. The model has achieved 100% accuracy in 80%, 60%, and 50% training categories, whereas in the case of 40% and 20% training data achieved 99.80% and 99.60% accuracy respectively. The overall accuracy is 99.95%. Fig. 18 illustrates the accuracy and loss of our model.

The accuracy of both training and testing is given in a blue dotted line and the loss is given in the red color line.

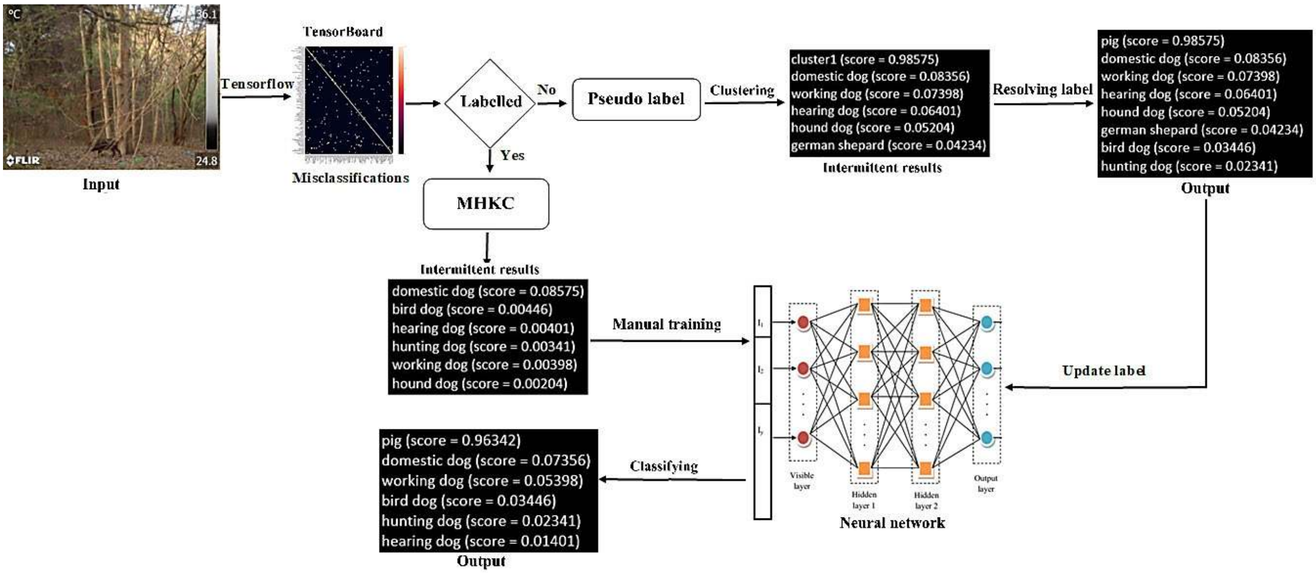

°C
36.1
24.8
FLIR
Input
Tensorflow
TensorBoard
Misclassifications
Labelled
No
Yes
Pseudo label
Clustering
cluster1 (score = 0.98575)
domestic dog (score = 0.08356)
working dog (score = 0.07398)
hearing dog (score = 0.06401)
hound dog (score = 0.05204)
german shepard (score = 0.04234)
Intermittent results
Resolving label
pig (score = 0.98575)
domestic dog (score = 0.08356)
working dog (score = 0.07398)
hearing dog (score = 0.06401)
hound dog (score = 0.05204)
german shepard (score = 0.04234)
bird dog (score = 0.03446)
hunting dog (score = 0.02341)
Output
MHKC
Intermittent results
domestic dog (score = 0.08575)
bird dog (score = 0.00446)
hearing dog (score = 0.00401)
hunting dog (score = 0.00341)
working dog (score = 0.00398)
hound dog (score = 0.00204)
Manual training
Update label
Visible layer
Hidden layer 1
Hidden layer 2
Output layer
Neural network
Classifying
pig (score = 0.96342)
domestic dog (score = 0.07356)
working dog (score = 0.05398)
bird dog (score = 0.03446)
hunting dog (score = 0.02341)
hearing dog (score = 0.01401)
Output

FIGURE 17. MHKC vs. Pseudo-labels on handling misclassifications.

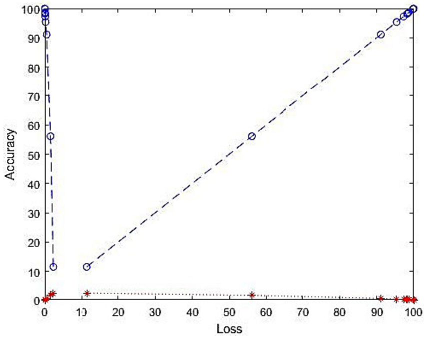


FIGURE 18. Accuracy–loss curve of MP-CNN (best viewed in color).

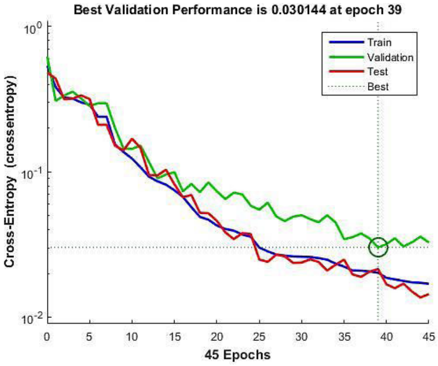


FIGURE 19. Performance plot of MP-CNN.

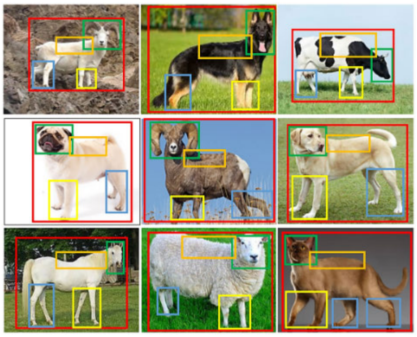


FIGURE 20. Sample results of our proposed MP-CNN model. The results depict both the object localization (in red color box) and the discriminative candidate parts.

The testing accuracy increases as the models get trained. The vertical blue line represents the training accuracy and the other one represents the testing accuracy line. The Training class is presented to the network during training, and the network is adjusted according to its error. Validation class is used to measure network generalization, and to halt training when generalization stops improving.

The testing class does not affect training and so it provides an independent measure of network performance during and after training. Fig. 19 indicates the best performance at epoch 39. The performance started saturating after epoch 39 and the parameters were dynamically changed during training.

MP-CNN localizes the discriminative parts (head, upper body, front leg, and hind leg) of the animals along with the object localization and a sample is depicted in Fig. 20. The model was able to detect the parts of animals even when there was more than one instance of animal in the image.

The performance of the various schemes is discussed in Table 4. MP-CNN has the best performance among all and Holistic CNN is close to MP-CNN but the difference in performance is not negligible as we are working with the animal classification system, where a wild animal cannot be classified as a domestic animal, due to its severity.

VII. COMPARATIVE STUDY

In this section, to evaluate the performance of the proposed system, we compared it with various related state–of –the–art systems namely deformable models and Deep Neural Net-

TABLE 4. Performance comparison of INCEPTION-V3, CNN and MP-CNN.

Metrics		Inception-V3	Holistic CNN	MP-CNN
Accuracy	80:20	0.988	0.988	0.998
	60:40	0.980	0.983	0.990
	50:50	0.973	0.981	0.983
	40:60	0.968	0.972	0.970
	20:80	0.962	0.969	0.964
Specificity	80:20	0.962	0.975	0.964
	60:40	0.942	0.963	0.967
	50:50	0.922	0.926	0.932
	40:60	0.902	0.921	0.932
	20:80	0.882	0.892	0.898
Precision	80:20	0.985	0.953	0.995
	60:40	0.975	0.981	0.977
	50:50	0.965	0.973	0.967
	40:60	0.960	0.962	0.968
	20:80	0.952	0.952	0.957
Accuracy		0.974	0.978	**0.981**
Specificity		0.922	**0.941**	0.929
Precision		0.967	0.969	**0.971**

TABLE 5. Summary of related works.

Related works	Approach	Data set	Accuracy	Application
[24]	Very deep convolutional neural networks –Residual network	Snapshot Serengeti dataset, 26 classes	88.9% accuracy in Top-1 and 98.1% in Top-5	Animal species identification
[22]	SIFT and cLBP	7,000 camera trap images of 18 species	82%	Automated species identification
[23]	Deep convolutional neural network	20 species with 14346 images.	38.31%	Animal species recognition
[25]	Deep neural network	Snapshot Serengeti dataset, 48 classes	92%	Animal species identification
[28]	Landmark-based shape representation with Grassmann Manifold	133 breeds with 8351 images	96.5%	Dog breed categorization
[27]	CNNs based on LeNet and GoogLeNet architectures	120 classes of dogs with 20580 images	LeNet (9.5%) GoogLeNet (8.9%)	Dog breed categorization
[29]	Exemplar-based geometric and appearance models of face parts.	133 dog breeds with 8351 images	67%	Dog Breed Classification

work architectures like VGG, ResNet, AlexNet, Inception, etc [47]. The summary of the compared methods is given in Table 5. In [24], the authors utilized the open dataset of Tanzania's Serengeti National Park, to classify the animals under 20 classes. They tested the dataset on various DNN architectures and concluded that ResNet topology was the best choice for classifying the Snapshot Serengeti dataset. In a similar work [25], the authors used DNN models to classify the animals from the same Serengeti dataset and achieved an accuracy of 92%. Their model was trained to classify the images only when it is confident about it. Other images were manually classified about the human. In yet another similar attempt [24], the authors classified the 20 species of animals using Deep Convolutional Neural Network, but their models' accuracy was far from the desired rate. In [22], the authors utilised the SIFT features along with that of Local Binary Pattern features to classify the 7,000 camera trap images into 18 species of animals. Their model achieved 82% accuracy. In [27], the author focused on classifying the various dog breeds using CNN. Their model classified 120 classes of dog breeds and achieved better accuracy with LeNet and GoogleNet. In [28] and [29], the authors classified the 133 classes of dog breeds using part based model. While, [28] used the landmark-based part model; [29] used the geometric of the face part to classify the animals.

For a fair comparison, we have implemented the selected methods given in [22]–[25], [27]–[29] with our dataset and the results are given in Table 5. Furthermore, we have compared our proposed MP-CNN with that of existing DNN architectures.

A. PROPOSED MODEL VS. OTHERS

The proposed model has been compared with a few related systems in various aspects. The model presented in [22], [24], [25] is about animal species detection following a supervised approach. A specific category [27]–[29] focuses on dog breed classification. Furthermore, [27], [29] are based on part models. Our approach is based on the Semi-supervised Multi-part CNN model. Excluding the retraining part, our model can be categorized as a supervised learning model.

Table 6 clearly shows the accuracy, specificity, and precision of various models along with our proposed approach. The best results in each category are highlighted. The proposed model has achieved the top results in most of the categories. Our proposed system works equally well even when the model is trained only with 20% of data. The

TABLE 6. **Proposed model vs. other related systems.**

Metrics	Dataset (Train : Test)	Supervised learning				Part based learning		MP-CNN
		Norouzzadeh [24]	Yu [22]	Gomez [25]	Liu [28]	Hsu [27]	Gao [29]	Our approach
Accuracy	80:20	0.87	0.93	0.85	0.83	0.85	0.87	0.99
	60:40	0.90	0.94	0.97	0.80	0.88	0.86	0.96
	50:50	0.89	0.96	0.88	0.77	0.87	0.88	0.97
	40:60	0.92	0.87	0.91	0.78	0.93	0.92	0.99
	20:80	0.89	0.85	0.88	0.71	0.87	0.92	0.96
Specificity	80:20	0.85	0.91	0.79	0.87	0.80	0.80	0.98
	60:40	0.88	0.94	0.78	0.86	0.85	0.83	0.93
	50:50	0.87	0.93	0.83	0.88	0.88	0.86	0.93
	40:60	0.91	0.89	0.87	0.90	0.87	0.89	0.91
	20:80	0.87	0.83	0.83	0.92	0.92	0.92	0.94
Precision	80:20	0.84	0.90	0.81	0.80	0.83	0.91	0.99
	60:40	0.87	0.91	0.83	0.83	0.88	0.92	0.98
	50:50	0.86	0.93	0.84	0.86	0.91	0.93	0.97
	40:60	0.89	0.84	0.88	0.89	0.94	0.97	0.96
	20:80	0.86	0.82	0.84	0.92	0.95	0.93	0.95

results of [25] were the next best to ours but their model was completely supervised and the results were not good for 20% and 40% training data categories. The performance of our model tends to improve as the model gets trained further for new classes of animals.

B. PROPOSED MP-CNN VS. EXISTING DNN MODELS

Animal image classification has seen many neural network architectures and each of the models has produced a different result. In this section, we compare our results with 6 other existing models [47] namely AlexNet, GoogleNet, ResNet 50, ResNet 101, VGG and Inception V3. ResNet also known as Residual Network, has won the 2016 ImageNet competition. It has different versions based on the number of layers it has. AlexNet is the winning architecture of the 2012 ILSVRC completion. The Tensorflow version of Inception V3 architecture is the first runner up in the ImageNet Large Visual Recognition Challenge. GoogleNet has 12 times lesser parameters than AlexNet, yet it is computationally efficient than AlexNet.

Fig.21 above illustrates the performance of different neural network architectures. Our dataset has been tested with all the above architectures and the results are depicted above. For our dataset, VGG has produced a fairly closer result than ours. Despite having 101 layers, ResNet 101 architecture produced an accuracy lower than Inception V3, which has only 9 layers. This shows that the number of layers is not an important factor in developing a neural network. The parameter setting plays a key role in achieving good results. Table 7 discusses the execution time for each of the individual units of our proposed model along with the performance of VGG architecture, which was found to be best among all other DNN architectures.

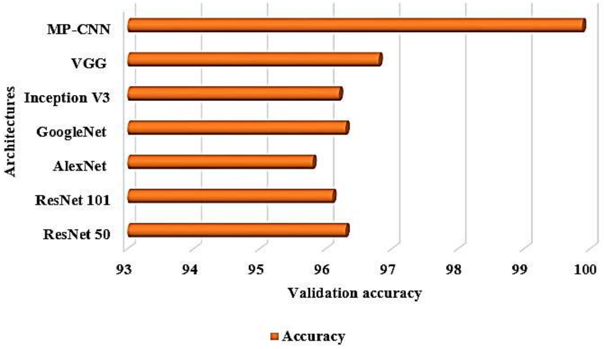


FIGURE 21. **Comparing the accuracy of various models.**

TABLE 7. **Execution time of proposed schemes vs. Existing technique [8].**

Factors	Our approach					DNN
	Individual results (in min)				Coalesced results	VGGNet [8]
	Tensorflow	MHKC	Pseudo label	MP-CNN	Semi-supervised learning based MP-CNN (in min)	
Pre-processing time/image	0.3	0.2	0.4	0.3	1.2	1.37
Training time/image	0.6	0.5	0.6	0.5	2.2	1.58
Processing time/image	0.3	0.3	0.4	0.2	0.8	1.42

Table.7 shows the performance comparison of various schemes. Our proposed approach has a very low processing time of 0.8 min per image. Any real-time monitoring

TABLE 8. Comparison with the state-of-the-art methods on the Oxford-IIIT Pet Dataset.

Method	Training annotation		Testing annotation		Accuracy	CNN feature
	Object	Parts	Object	Parts		
Our approach	-	-	-	-	98.17	CNN
InterActive [13]	-	-	-	-	93.45	VGGNet
TL Atten [14]	-	-	-	-	92.51	VGGNet
CAN [15]	-	-	-	-	81.70	VGGNet
FOAF [17]	-	-	-	-	91.39	VGGNet
Bilinear CNN [16]	-	-	-	-	84.10	VGG-M
DGM [18]	-	-	-	-	60.19	GoogleNet
Coarse to Fine [10]	✓	-	✓	-	82.50	VGGNet
PG Alignment [12]	✓	-	✓	-	82.80	VGGNet
Deep LAC [8]	✓	✓	✓	-	84.10	AlexNet
Part based R-CNN [7]	✓	✓	✓	✓	76.37	AlexNet
PS-CNN [9]	✓	✓	✓	-	76.20	AlexNet
PN-DCN [6]	✓	✓	✓	✓	85.40	AlexNet
Webly-supervised [11]		✓	-	-	78.60	AlexNet
HPM [4]	✓	✓	✓	✓	73.30	CNN
POOF [5]	✓	✓	✓	✓	66.35	CNN

application will require a faster response. Our coalesced approach for animal image classification has a very quick response time and a higher accuracy for classification when compared to other approaches.

C. PROPOSED MP-CNN VS. STATE-OF-THE-ART METHODS ON OXFORD IIIT PET DATASET

The proposed MP-CNN model is tested on the benchmark Oxford-IIIT Pet dataset [48]. The dataset consists of 37 different pet categories of which 25 are dog breeds and 12 are cat breeds. It has a total of 3680 training images and 3669 test images. Table 8 compares the proposed MP-CNN with various state-of-the-art methods on the Oxford IIIT Pet dataset. The details of training and testing annotations on both object and part level is also given.

The proposed methodology achieved better accuracy on the Oxford IIIT Pet dataset when compared to the state-of-the-art methods. Further, we have not employed annotations or bounding boxes for localizing the object or parts. InterActive model and FOAF achieved closer results and both do not employ annotations.

The performance of individual components in MP-CNN is tested on the Oxford IIIT Pet dataset. From Table 9, it could be inferred that the Oxford IIIT pet dataset performed better with object level localization than part selection. In converse, our dataset performed better with part level than the object level.

TABLE 9. Performance of components in MP-CNN on our dataset and Oxford IIIT PET dataset.

Method	Accuracy (%)	
	Our dataset	Oxford IIIT Pet dataset
Object level	91.85	94.23
Part level	94.54	89.58
Object +Part level	99.95	98.17

The contradiction arises mainly due to the way the objects are localized initially. The training images of Oxford Pet dataset focuses more on the foreground object and the background clutter is very minimal, leading to more accurate localization of the object. Nevertheless, on both the dataset, a combination of the object-part model had higher performance.

VIII. CONCLUSION AND FUTURE SCOPE

Tensorflow is a novel machine learning software library from Google's Brain. It is well suited for the automatic classification of images despite the number of training images. In this paper, we focused on classifying 27 classes of animals with 35,992 training images. In summary, we were able to classify the 27 classes of animals with the highest accuracy of about 96% and 98% in category I and V dataset respectively with Tensorflow. We have further worked on reducing the misclassification rate by applying Modified Hellinger Kernel Classifier to the training dataset of misclassified categories. This approach has further increased the training accuracy to about 99.52% of the overall model. Furthermore, we utilized the semi-supervised learning based pseudo-labels to handle any new classes of images with no ground truth. This is one of the crucial requirements for an automated real-time system. For a fine-grained animal breed classification, we utilize the MP-CNN, that has been tailored for our dataset and with which we improved the accuracy to about 99.95%.

In particular, we are interested in working with a wild animal detection system for monitoring its moment in the residential area, thereby alerting the residents. As part of our future work, we plan to embrace the images of the animal in various positions (facing away from the camera) under various lighting conditions (day and night) to further increase the stiffness of the training dataset. Also, we plan to extend the proposed model to work with thermal images. As part of our work, we will release a new dataset of various animals' infrared images.

Acknowledgment

The authors would like to thank the anonymous reviewers for their constructive comments and valuable suggestions, which greatly helped us to improve the technical quality and presentation of the manuscript. They would also like to thank VIT for providing VIT seed grant for purchasing the necessary tools for the research.

REFERENCES

[1] A. Ahmeda, H. Yousifa, R. Kaysb, and Z. Hea, "Semantic region of interest and species classification in the deep neural network feature domain," *Ecological Inform.*, vol. 52, Jul. 2019, pp. 57–68.

[2] B. Kong, J. Supančič, D. Ramanan, and C. C. Fowlkes, "Cross-domain image matching with deep feature maps," *Int. J. Comput. Visions*, vol. 127, no. 365, pp. 1–13, Jan. 2019.

[3] M. Zhanyu, Y. Ding, S. Wen, J. Xie, Y. Jin, Z. Si, and H. Wang, "Shoe-print image retrieval with multi-part weighted CNN," *IEEE Access*. vol. 7, pp. 59728–59736, 2019.

[4] L. Xie, Q. Tian, R. Hong, S. Yan, and B. Zhang, "Hierarchical part matching for fine-grained visual categorization," in *Proc. Int. Conf. Comput. Vis. (ICCV)*, Dec. 2013, pp. 1641–1648.

[5] T. Berg and P. Belhumeur, "Poof: Part-based one-vs.-one features for fine-grained categorization, face verification, and attribute estimation," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2013, pp. 955–962.

[6] S. Branson, G. V. Horn, S. Belongie, and P. Perona, "Bird species categorization using pose normalized deep convolutional Nets," 2014, *arXiv:1406.2952*. [Online]. Available: https://arxiv.org/abs/1406.2952

[7] N. Zhang, J. Donahue, R. Girshick, and T. Darrell, "Part-based R-CNNs for fine-grained category detection," in *Proc. Eur. Conf. Comput. Vis. (ECCV)*, 2014, pp. 834–849.

[8] D. Lin, X. Shen, C. Lu, and J. Jia, "Deep LAC: Deep localization, alignment and classification for fine-grained recognition," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2015, pp. 1666–1674.

[9] S. Huang, Z. Xu, D. Tao, and Y. Zhang, "Part-stacked CNN for fine-grained visual categorization," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2016, pp. 1173–1182.

[10] H. Yao, S. Zhang, Y. Zhang, J. Li, and Q. Tian, "Coarse-to-fine description for fine-grained visual categorization," *IEEE Trans. Image Process.*, vol. 25, no. 10, pp. 4858–4872, Oct. 2016.

[11] Z. Xu, S. Huang, Y. Zhang, and D. Tao, "Webly-supervised fine-grained visual categorization via deep domain adaptation," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 40, no. 5, pp. 1100–1113, May 2018.

[12] Z. Xu, D. Tao, S. Huang, and Y. Zhang, "Friend or foe: Fine-grained categorization with weak supervision," *IEEE Trans. Image Process.*, vol. 26, no. 1, pp. 135–146, Jan. 2017.

[13] L. Xie, Q. Tian, M. Wang, and B. Zhang, "Spatial pooling of heterogeneous features for image classification," *IEEE Trans. Image Process.*, vol. 23, no. 5, pp. 1994–2008, May 2013.

[14] J. Krause, H. Jin, J. Yang, and L. Fei-Fei, "Fine-grained recognition without part annotations," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2015, pp. 5546–5555.

[15] M. Simon and E. Rodner, "Neural activation constellations: Unsupervised part model discovery with convolutional networks," in *Proc. Int. Conf. Comput. Vis. (ICCV)*, Dec. 2015, pp. 1143–1151.

[16] T.-Y. Lin, A. RoyChowdhury, and S. Maji, "Bilinear CNN models for fine-grained visual recognition," in *Proc. Int. Conf. Comput. Vis. (ICCV)*, Dec. 2015, pp. 1449–1457.

[17] X. Zhang, H. Xiong, W. Zhou, and Q. Tian, "Fused one-vs-all features with semantic alignments for fine-grained visual categorization," *IEEE Trans. Image Process.*, vol. 25, no. 2, pp. 878–892, Feb. 2016.

[18] L. Zhang, Y. Yang, M. Wang, R. Hong, L. Nie, and X. Li, "Detecting densely distributed graph patterns for fine-grained image categorization," *IEEE Trans. Image Process.*, vol. 25, no. 2, pp. 553–565, Feb. 2016.

[19] H. Zheng, J. Fu, T. Mei, and J. Luo, "Learning multi-attention convolutional neural network for fine-grained image recognition," in *Proc. Int. Conf. Comput. (ICCV)*, Oct. 2017, pp. 5209–5217.

[20] C. Huang, H. Li, Y. Xie, Q. Wu, and B. Luo, "PBC: Polygon-based classifier for fine-grained categorization," *IEEE Trans. Multimedia*, vol. 19, no. 4, pp. 673–684, Apr. 2017.

[21] T. Xiao, Y. Xu, K. Yang, J. Zhang, Y. Peng, and Z. Zhang, "The application of two-level attention models in deep convolutional neural network for fine-grained image classification," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2015, pp. 842–850.

[22] X. Yu, J. Wang, R. Kays, P. A. Jansen, T. Wang, and T. Huang, "Automated identification of animal species in camera trap images.," *EURASIP J. Image Video Process.*, vol. 52, no. 1, p. 25, Dec. 2013. doi: 10.1186/1687-5281-2013-52.

[23] G. Chen, T. X. Han, Z. He, R. Kays, and T. Forrester, "Deep convolutional neural network based species recognition for wild animal monitoring," in *Proc. IEEE Int. Conf. Image Process. (ICIP)*, Oct. 2014, pp. 858–862. doi: 10.1109/ICIP.2014.7025172.

[24] M. S. Norouzzadeh, A. Nguyen, M. Kosmala, A. Swanson, M. S. Palmer, C. Packer, and J. Clune, "Automatically identifying, counting, and describing wild animals in camera-trap images with deep learning," *Proc. Nat. Acad. Sci. USA*, vol. 115, no. 25, pp. 5716–5725, Jun. 2018.

[25] A. G. Villa, A. Salazar, and F. Vargas, "Towards automatic wild animal monitoring: Identification of animal species in camera-trap images using very deep convolutional neural networks," *Ecological Informat.*, vol. 41, pp. 24–32, Sep. 2017. doi: 10.1016/j.ecoinf.2017.07.004.

[26] K. Figueroa, A. Camarena-Ibarrola, J. García, and H. T. Villela, "Fast Automatic Detection of Wildlife in Images from Trap Cameras," in *Iberoamerican Congress on Pattern Recognition*. Berlin, Germany: Springer, 2014, pp. 940–947. doi: 10.1007/978-3-319-12568-8_114.

[27] D. Hsu, "Using convolutional neural networks to classify dog breeds," CS231n, Convolutional Neural Netw. Vis. Recognit., Stanford Univ., Stanford, CA, USA, 2015. [Online]. Available: http://cs231n.stanford.edu/reports/2015/pdfs/fcdh_FinalReport.pdf

[28] J. Liu, A. Kanazawa, D. Jacobs, and P. Belhumeur, "Dog breed classification using part localization," in *Proc. 12th Eur. Conf. Comput. Vis. (ECCV)*, A. Fitzgibbon, S. Lazebnik, P. Perona, Y. Sato, and C. Schmid, Eds. Berlin, Germany: Springer-Verlag, 2012, pp. 172–185.

[29] S. Gao, I. W.-H. Tsang, and Y. Ma, "Learning category-specific dictionary and shared dictionary for fine-grained image categorization," *IEEE Trans. Image Process.*, vol. 23, no. 2, pp. 623–634, Feb. 2014.

[30] P. Goldsborough. (2016). *A Tour of TensorFlow*. [Online]. Available: https://www.researchgate.net/publication/308895905_A_Tour_of_TensorFlow

[31] J. Heaton, S. McElwee, J. Fraley, J. Cannady, "Early stabilizing feature importance for TensorFlow deep neural networks," in *Proc. Int. Joint Conf. Neural Netw.(IJCNN)*, May 2017, pp. 4618–4624. doi: 10.1109/IJCNN.2017.7966442.

[32] H. Yi, H. Jung, and S. Bae, "Deep neural networks for traffic flow prediction," in *Proc. IEEE Int. Conf. Big Data Smart Comput. (BigComp)*, Feb. 2017, pp. 328–331. doi: 10.1109/BIGCOMP.2017.7881687.

[33] Y. Chen, S. Duffner, A. Stoian, J. Y. Dufour, and A. Baskurt, "Pedestrian attribute recognition with part-based CNN and combined feature representations," in *Proc. 15th Int. Conf. Comput. Vis. Theory Appl. (VISAPP)*, Funchal, Portugal, Jan. 2018, pp. 114–122.

[34] A. Iscen, G. Tolias, P.-H. Gosselin, and H. Jégou, "A comparison of dense region detectors for image search and fine-grained classification," *IEEE Trans. Image Process.*, vol. 24, no. 8, pp. 2369–2381, Aug. 2015.

[35] P. F. Felzenszwalb, R. B. Girshick, D. McAllester, and D. Ramanan, "Object detection with discriminatively trained part-based models," in *Proc. IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 32, no. 9, pp. 1627–1645, Sep. 2010.

[36] K. Duan, D. Parikh, D. Crandall, and K. Grauman, "Discovering localized attributes for fine-grained recognition," in *Proc. IEEE Int. Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2012, pp. 3474–3481.

[37] S. Yang, L. Bo, J. Wang, and L. G. Shapiro, "Unsupervised template learning for fine-grained object recognition," in *Proc. Adv. Neural Inf. Process. Syst. (NIPS)*, 2012, pp. 3122–3130.

[38] S. Lapuschkin, A. Binder, G. Montavon, K.-R. Müller, and W. Samek, "Analyzing classifiers: Fisher vectors and deep neural networks," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2016, pp. 2912–2920.

[39] J. Masci, U. Meier, D. Cire an, and J. Schmidhuber, "Stacked convolutional auto-encoders for hierarchical feature extraction," in *Artificial Neural Networks and Machine Learning—ICANN* (Lecture Notes in Computer Science), T. Honkela, W. Duch, M. Girolami, and S. Kaski, eds., vol. 6791. Berlin, Germany: Springer, 2011.

[40] P.-H. Gosselin, N. Murray, H. Jégou, and F. Perronnin, "Revisiting the Fisher vector for fine-grained classification," *Pattern Recognit. Lett.*, vol. 49, pp. 92–98, Nov. 2014.

[41] S. D. Meena and L. Agilandeeswari, "Stacked convolutional autoencoder for detecting animal images in cluttered scenes with a novel feature extraction framework," in *Proc. 8th Int. Conf. Soft Comput. Problem Solving (SocProS)*, 2018, pp. 1–9.

[42] J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, and L. Fei-Fei, "ImageNet: A large-scale hierarchical image database," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2009, pp. 248–255.

[43] N. Zhang, R. Farrell, F. Iandola, and T. Darrell, "Deformable part descriptors for fine-grained recognition and attribute prediction," in *Proc. ICCV*, Dec. 2013, pp. 729–736.

[44] J. R. Uijlings, K. E. van de Sande, T. Gevers, and A. W. Smeulders, ''Selective search for object recognition,'' *Int. J. Comput. Vis.*, vol. 140, no. 2, pp. 154–171, Sep. 2013.

[45] F. systems. *Thermal Imaging, Night Vision and Infrared camera system*. Accessed: Jan. 3, 2019. [Online]. Available: https:www.flir.com

[46] O. Russakovsky, J. Deng, H. Su, J. Krause, S. Satheesh, S. Ma, Z. Huang, A. Karpathy, A. Khosla, M. Bernstein, A. C. Berg, and L. Fei-Fei, ''ImageNet large scale visual recognition challenge,'' *Int. J. Comput. Vis.*, vol. 115, no. 3, pp. 211–252, Dec. 2015.

[47] S. Bahrampour, N. Ramakrishnan, L. Schott, and M. Shah. (2016). *Comparative Study of Deep Learning Software Frameworks*. [Online]. Available: https://www.researchgate.net/publication/284476468_Comparative_Study_of_Caffe_Neon_Theano_and_Torch_for_Deep_Learning

[48] O. M. Parkhi, A. Vedaldi, A. Zisserman, and C. V. Jawahar, ''Cats and dogs,'' in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2012, pp. 3498–3505.

S. DIVYA MEENA received the B.Tech. degree in information technology from the Vellore Institute of Technology (VIT), Vellore, India, in 2014, and the M.E. degree in computer science and engineering from Anna University, Chennai, in 2016. She is currently pursuing the Ph.D. degree with the School of Information Technology and Engineering, VIT. From 2016 to 2017, she was an Assistant Professor with the Jansons Institute of Technology, Coimbatore, India. Her research interests include machine learning, neural networks, and image processing and fuzzy logic.

L. AGILANDEESWARI received the bachelor's degree in information technology and the master's degree in computer science and engineering from Anna University, in 2005 and 2009, respectively, and also completed the Ph.D. degree.

She is currently the HOD and an Associate Professor with the Department of Digital Communications, School of Information Technology and Engineering (SITE), Vellore Institute of Technology (VIT), Vellore. She is having around 13+ years of teaching experience. She has published 50+ articles in peer-reviewed reputed journals. She also published about 13 engineering books as per Anna University syllabus. She is a Life Time Member of the Computer Society of India. Her reputed publications include research articles in peer-reviewed journals namely *Expert Systems with Applications*, the *Journal of Ambient Intelligence and Humanized Computing*, *Multimedia Tools and Applications*, and the *Journal of Applied Remote Sensing* indexing at Thomson Reuters with an average impact factor of five. Her areas of interests include image and video watermarking, image and video processing, neural networks, Fuzzy logic, machine learning, cryptography, the IoT, information centric networks, and remote sensing. She has also got the Best Researcher Award for the past four years, from 2015 to 2019. She is a peer Reviewer of journals including the IEEE Access, *Array*, *Artificial Intelligence Review*, *Informatics in Medicine Unlocked*, *Neuro Computing*, *Computers and Electrical Engineering*, the *Journal of King Saud University-Computer and Information Sciences*, *IET Review*, and the *Journal of Engineering Science and Technology*.