Received May 2, 2019, accepted May 19, 2019, date of publication May 22, 2019, date of current version June 4, 2019.

Digital Object Identifier 10.1109/ACCESS.2019.2918274

Bird Image Retrieval and Recognition Using a Deep Learning Platform

YO-PING HUANG[1,2], (Senior Member, IEEE), AND HAOBIJAM BASANTA[1]
[1]Department of Electrical Engineering, National Taipei University of Technology, Taipei 10608, Taiwan
[2]Department of Computer Science and Information Engineering, National Taipei University, New Taipei City 23741, Taiwan

Corresponding author: Yo-Ping Huang (yphuang@ntut.edu.tw)

This work was supported in part by the Ministry of Science and Technology, Taiwan, under Grant MOST108-2321-B-027-001 and Grant MOST107-2221-E-027-113.

ABSTRACT Birdwatching is a common hobby but to identify their species requires the assistance of bird books. To provide birdwatchers a handy tool to admire the beauty of birds, we developed a deep learning platform to assist users in recognizing 27 species of birds endemic to Taiwan using a mobile app named the Internet of Birds (IoB). Bird images were learned by a convolutional neural network (CNN) to localize prominent features in the images. First, we established and generated a bounded region of interest to refine the shapes and colors of the object granularities and subsequently balanced the distribution of bird species. Then, a skip connection method was used to linearly combine the outputs of the previous and current layers to improve feature extraction. Finally, we applied the softmax function to obtain a probability distribution of bird features. The learned parameters of bird features were used to identify pictures uploaded by mobile users. The proposed CNN model with skip connections achieved higher accuracy of 99.00 % compared with the 93.98% from a CNN and 89.00% from the SVM for the training images. As for the test dataset, the average sensitivity, specificity, and accuracy were 93.79%, 96.11%, and 95.37%, respectively.

INDEX TERMS Bird image recognition, convolutional neural network, deep learning, mobile app.

I. INTRODUCTION

The everyday pace of life tends to be fast and frantic and involves extramural activities. Birdwatching is a recreational activity that can provide relaxation in daily life and promote resilience to face daily challenges. It can also offer health benefits and happiness derived from enjoying nature [1].

Numerous people visit bird sanctuaries to glance at the various bird species or to praise their elegant and beautiful feathers while barely recognizing the differences between bird species and their features. Understanding such differences between species can enhance our knowledge of exotic birds as well as their ecosystems and biodiversity. However, because of observer constraints such as location, distance, and equipment, identifying birds with the naked eye is based on basic characteristic features, and appropriate classification based on distinct features is often seen as tedious. In the past, computer vision [2], [3] and its subcategory of recognition, which use techniques such as machine learning, have been extensively researched to delineate the specific features of objects, including vegetables and fruits [4], landmarks [5], clothing [6], cars [7], plants [8], and birds [9], within a particular cluster of scenes. However, considerable room for improvement remains in the accuracy and feasibility of bird feature extraction techniques. Detection of object parts is challenging because of complex variations or similar subordinate categories and fringes of objects. Intraclass and interclass variation in the silhouettes and appearances of birds is difficult to identify correctly because certain features are shared among species.

To classify the aesthetics of birds in their natural habitats, this study developed a method using a convolutional neural network (CNN) to extract information from bird images captured previously or in real time by identifying local features. First, raw input data of myriad semantic parts of a bird were gathered and localized. Second, the feature vectors of each generic part were detected and filtered based on shape, size, and color. Third, a CNN model was trained with the bird pictures in a graphics processing unit (GPU) for feature vector extraction with consideration of the aforementioned characteristics, and subsequently the classified, trained data were stored on a server to identify a target object.

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

Ultimately, information obtained from a bird image uploaded by an end-user, captured using a mobile camera, can be navigated through the client–server architecture to retrieve information and predict bird species from the trained model stored on the server. This process facilitates efficient correlation of fine-grained object parts and autonomous bird identification from captured images and can contribute considerable, valuable information regarding bird species.

The remainder of this paper is organized as follows. Section II briefly reviews related approaches for fine-grained visual categorization. Section III describes the various types of dataset used for feature extraction. Section IV focuses on the deep learning model and its features used in object part models, and describes the correlation between part localization and fine-grained feature extraction. Section IV also describes various correlation requirements, such as data augmentation, for excellent performance, localization, segmentation, identification of subcategories, as well as the requirement of a classifier for effective object prediction. The experimental results and analysis of the datasets are presented in Section V. Section VI summarize the discussion and limitation part of the study. Conclusions and directions for future study are provided in Section VII.

II. RELATED WORK

Recently, some fine-grained visual categorizations methods have been proposed for species identification, and they have become a promising approach within computer vision research, with applications in numerous domains [10]. Numerous fine-grained recognition datasets, such as ImageNet, ILSVRC, Caltech-256, and CUB 200, have trained models with a wide variety of data to extract global features such as colors, textures, and shapes from multilabel objects [11]. Many approaches have been applied for generic object recognition [12]. Some methods apply local part learning that uses deformable part models and region-CNN for object detection [13], generation of a bounding box, and selection of distinctive parts for image recognition. Some studies have focused on discriminative features based on the local traits of birds [14], [16]. Simultaneous detection and segmentation are used to localize score detections effectively [15]. Pose-normalization and model ensembles [16] are also used to improve the performance of fine-grained detection by generating millions of key point pairs through fully convolutional search. Discriminative image patches and randomization techniques are integrated to distinguish classes of images and prevent overfitting [17]. The present work also approached the learning of discriminative image features using a CNN architecture for fine-grained recognition. However, a complementary approach using domain knowledge of general bird features was integrated to provide detailed information about the predicted bird.

The advancement of consumer products, such as smartphones, digital cameras, and wearable gadgets [18], has transformed multidisciplinary approaches toward technology by connecting the physical and digital worlds. High-resolution digital cameras in smartphones are the most pervasive tools used for recognizing the salient features of physical objects, enabling users to detect, identify objects and share related knowledge. Birds present in a flock are often deeply colorful; therefore, identification at a glance is challenging for both birdwatchers and onlookers because of birds' ambiguous semantic features [19]. To address this problem, an information retrieval model for birdwatching has been proposed that uses deep neural networks to localize and clearly describe bird features with the aid of an Android smartphone [20], [21].

III. DATA ACQUISITION

Feature extraction is vital to the classification of relevant information and the differentiation of bird species. We combined bird data from the Internet of Birds (IoB) and an Internet bird dataset to learn the bird species.

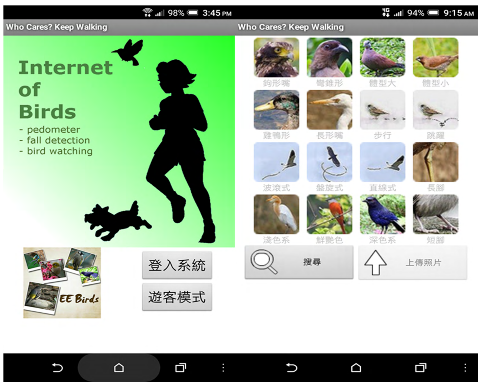


FIGURE 1. IoB interface.

A. IOB

The IoB is a crowdsourced metasearch-engine database specifically for birds, where any individual can store bird images and instantly retrieve information about the birds therein. Uploaded bird images are identified from extracted features. This platform encourages individuals to become involved in birdwatching and to enrich their knowledge of various bird species. The IoB is available online for free (with keyword: Who Cares? Keep Walking). Fig. 1 shows the app interface. Because a fall detection module is embedded in the system, the app also serves as a wellness platform to assist individuals in staying safe while birdwatching. In addition, the system can track the distance individuals cover from their daily physical strides using a pedometer to promote fitness and motivate users to walk while birdwatching [22].

B. INTERNET BIRD IMAGES

A pool of images is required for deep learning of subcategorization. Bird images containing 27 bird species endemic to Taiwan on various backgrounds were compiled from the IoB and several other online resources. The use of public-domain images has benefits and drawbacks. Although Internet image sources add diversity to the dataset, the images may be contaminated with noise, harshness, spurious pixels, and blurred parts, all of which degrade image quality. Therefore, to limit the intensity of deformity in an assortment of images, high-pixel images with clear boundaries were used. Finally, to obtain standardized balance in the dataset, the bird species images were transformed and augmented as follows [23]:

- Random flipping: Images were horizontally and vertically flipped.
- Rotation: Images were randomly rotated (maximum angle of 25°) for training.
- Translation: Images were randomly shifted −10 to 10pixels.
- Zero-phase component analysis whitening: Dimension and redundancy in the matrix of pixel images were decreased.
- Gaussian filtering: Images were blurred for effective smoothing of noise.

In deep learning algorithms, feature extraction is a generalization step to differentiate the learning categories of input data patterns.

Object recognition with a high-level feature extraction architecture comprises the following steps: (1) data content analysis, in which all generic raw data are preprocessed to extract nonlinear transformations and to fit the parameters into a machine learning model for feature extraction; (2) optimal probabilities of relevant structural information from each tuned parameter are clubbed into a new array of classifiers; and (3) a prediction is made based on trained and learned parameters. To extract multiple feature levels from raw data and evaluate the performance of the CNN for the dataset [24]–[26], the dataset was split into the three modules discussed as follows. (1) The training dataset comprised raw data samples that were incorporated into the training model to determine specific feature parameters, perform correlational tasks, and create a related classification model. (2) The validation dataset was used to tune hyperparameters of the trained model to minimize overfitting and validate performance. The model regularizes early stopping to prevent overfitting and to enhance learning when the precision of the training dataset increases while the error of the validation dataset remains the same or decreases. (3) The test dataset was used to test the classifier parameters and assess the performance of the actual prediction of the network model. Once the features had been extracted from the raw data, the trained prediction model was deployed to classify new input images. Fig. 2 shows the module for extracting unique features of birds with the CNN and predicting the most classified labels for the input images.

Table 1 provides a list of terms and related abbreviations commonly used in this study.

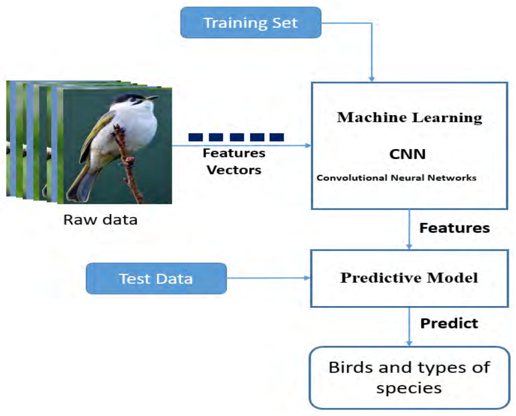


FIGURE 2. Feature extraction paradigm for bird images.

TABLE 1. List of terms and abbreviations.

Terms	Abbreviation
Internet of birds	IoB
Convolution neural network	CNN
Support vector machine	SVM
Batch normalization	BN
Rectified linear unit	ReLU
Fully connected	FC
Central processing unit	CPU
Graphic processing unit	GPU
Tensor flow	TF
Software development kit	SDK
Native development kit	NDK
Java native interface	JNI
Application programming interface	API
Hypertext transfer protocol secure	HTTPS
True positive	TP
False positive	FP
True negative	TN
False negative	FN

IV. PROPOSED DEEP LEARNING MODEL

The emergence of deep learning [27] algorithms has resulted in highly complex cognitive tasks for computer vision and image recognition. Recently, deep learning models have become the most popular tool for big data analysis and artificial intelligence [28], outperforming traditional image classification algorithms, and they are currently being downscaled for feasible mobile implementation. The proposed deep learning model for bird image classification using the CNN framework is described as follows.

A. CNN ARCHITECTURE

The model of CNN configuration for bird identification utilized a stack of convolutional layers comprising an input layer, two FC layers, and one final output softmax layer. Each convolutional layer comprised (a) 5 × 5 convolution, (b) BN, (c) ReLU activation, and (d) pooling layers. This section explains how to construct an optimized CNN model, why the parameters and hyperparameters must be tuned before training, the total number of convolutional layers, the size of the kernels for all relative convolutional layers, and

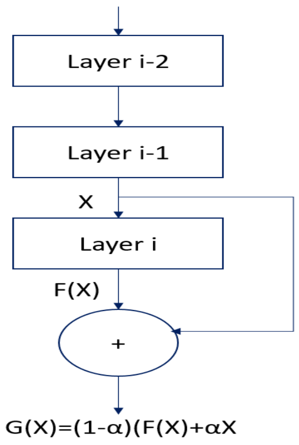


FIGURE 3. Framework of skip connections.

the likelihood of retaining the anode during dropout regularization for the dataset.

B. SKIP CONNECTIONS

When images are learned, deep neural network models train a base network from scratch to identify associations of features and patterns in the target dataset. Features are transformed from general to specific by the last layer of the network to predict the outputs of newly imposed inputs. If first-layer features are general and last-layer features are specific, then a transition from general to specific must have occurred somewhere in the network [29]. To address and quantify the degree to which a particular layer is general or specific, we proposed adding skip connections [30] among corresponding convolutional layers, as shown in Fig. 3. The skip layer connections should improve feature extraction through weighted summation of corresponding layers as follows:

$$G(X) = (1-\alpha)F(X) + \alpha X \tag{1}$$

where X is the input, $F(X)$ is a function of input X, $G(X)$ is a linear combination of $F(X)$ and X, and α is a weight in the unit interval [0,1]. To check specific layers, we used different weights. For example, if $\alpha > 0.5$, then result from the previous layer contributes less to overall performance than the layers preceding it. By contrast, if $\alpha < 0.5$, then result from the previous layer contributes more to the overall performance. Using these skip connections can facilitate network training by reducing memory usage and increasing performance by concatenating the feature maps of each convolution layer.

C. TRAINING OF THE BIRD DATASET

Learning of bird species by the CNN was implemented on a GPU workstation with a 12 Intel Xeon CPU, 32 GB of memory, and an Nvidia GeForce 2 11 GB GTX 1080 Ti graphics card on a TF platform [31]–[33]. During training, input color images with a fixed size of 112 × 112 pixels were fed into CNN for feature extraction and bird image recognition. This study uses a dataset comprising 3563 images of 27 bird species. The dataset was split into 2280 images for training, 570 for validation, and 713 for testing. The input images passed through a hierarchical stack of convolutional layers to extract distinct features, such as color, shape, and edges, with varying orientations of the head, body, legs, and tail shown in the images. The first convolutional layer transformed the input image into pixels, propelled it to the next layer, and followed the feature extraction procedure until the input image had been precisely classified with a probability distribution. To capture the features of the input image, every convolutional filter had a kernel size of 3 × 3 pixels and a high activation map that slid across the entire input volume. The stride was fixed at one by shifting the kernel one unit at a time to control the filter convolving around the input of the next pixel so that the output volume would not shrink and the yield would be an integer rather than a fraction; that is, $(i-k+2q)/(s+1)$, where i is the input height or length, k is the filter,q is the padding, and s is the stride. The padding was attuned to one around the input image to preserve the spatial resolution of output feature map after convolution; that is, $q = (k-1)/2$. Spatial pooling [34] was implemented to localize and separate the chunks of images with a 2 × 2 pixel window size, max pooling, and two strides, where the maximum pixel rate in each chunk of an image was considered. The stack of convolutional layers was followed by an element-wise activation function, the ReLU, to maintain a constant volume throughout the network.

To implement the skip connection in the network, downsampling is performed by conv3 and conv4 with a stride of 2. We directly use skip connection when the input and output have the same dimensions. When the dimensions of the output are increased, the shortcut performs identity mapping with an extra zero-padding entry for increasing dimensions. Two FC layers were implemented with the same 4096-dimension configuration to learn the gradient descent, compute the target class scores in the training set for each image, and localize objects positioned anywhere in the input image. A schematic of the ConvNet architecture is presented in Fig. 4, and the parameter configuration for ConvNet is provided in Table 2.

After the FC layers were added, the n-softmax layer activation function [35] was added; here, n is the number of bird categories. The softmax layer yields a probabilistic interpretation of multiple classes. Each label corresponds to the likelihood that the input images are correctly classified using vector-to-vector transformation, thereby minimizing cross-entropy loss.

$$\text{softmax } \sigma(x)_i = \frac{e^{x_i}}{\sum_{j=1}^{K} e^{x_j}}, \quad \textit{for } i = 1, \ldots, K. \tag{2}$$

where x_i is the ith element of the input vector $\boldsymbol{x}$, $\sum_i \sigma(x)_i = 1$ and $\sigma(x)_i > 0$, which is the probability distribution over a set of outcomes.

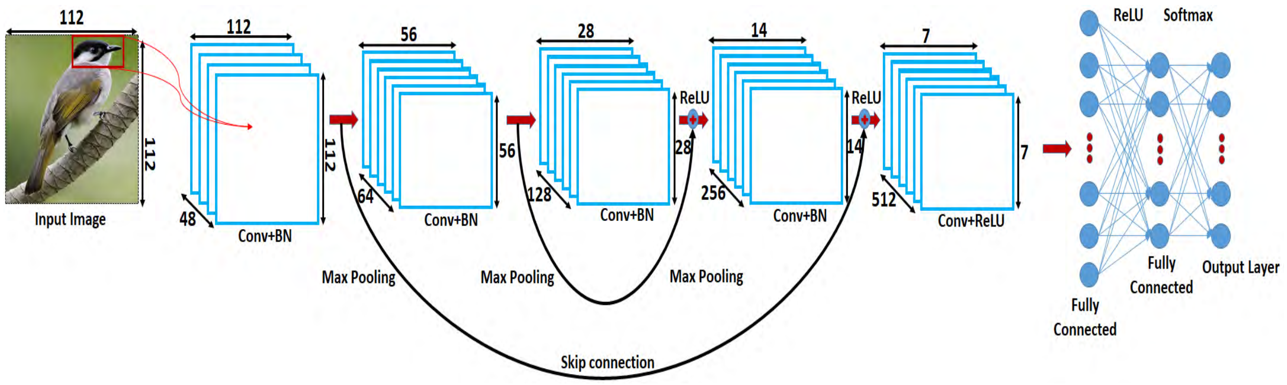


FIGURE 4. **CNN architecture for detecting bird images.**

TABLE 2. **Convnet parameter configuration for bird image detection system.**

Layer	Input size ($h \times w$)	Filter size ($h \times w$)	No. of filters	Pad ($h \times w$)	Stride
Input	112 ×112				
Conv+BN+ReLU	112 ×112	3 × 3	48	1 × 1	1
Max Pooling	56 × 56	2 × 2	48	-	2
Conv+BN+ReLU	56 × 56	3 × 3	64	1 × 1	1
Pooling	28 × 28	2 × 2	64	-	2
Conv+BN+ReLU	28 × 28	3 × 3	128	1 × 1	1
Max Pooling	14 × 14	2 × 2	128	-	2
Conv+BN+ReLU	14 × 14	3 × 3	256	1 × 1	1
Max Pooling	7 × 7	2 × 2	256	-	2
Conv+BN+ReLU	7 × 7	3 × 3	512	1 × 1	1
FC	4096	-	-	-	1
FC	4096	-	-	-	1
Softmax layer	27	-	-	-	1

** Conv = Convolution, BN = Batch normalization, FC = Fully connected, h = Height, w = Width.

D. FEATURE EXTRACTION

Extracting features from raw input images is the primary task when extracting relevant and descriptive information for fine-grained object recognition [36]–[38]. However, because of semantic and intraclass variance, feature extraction remains challenging. We separately extracted the features in relevant positions for each part of an image and subsequently learned the parts of the model features that were mapped directly to the corresponding parts. The features were calculated using ReLU 5 and ReLU 6. Localization was used to find object parts defined by bounding box coordinates and their dimensions (width and height) in the image [39]. For the localization task an intersection over union score >0.5 was set for our model. An FC layer with a ReLU was used to predict the location of bounding box B_x. Subsequent steps of the learning algorithm were for learning the map of the feature vectors of the input image, deciding whether the region fit an object class of interest, and then classifying the expected output with the correct labels in the image. For a given image, feature vectors represent the probability of target object centrality in

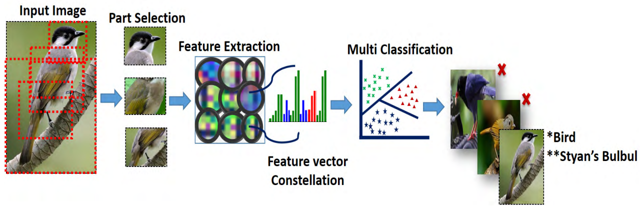


FIGURE 5. **Input raw data and feature illustration for a classifier.**

the database, and the softmax classifier produces the probability scores for each label. Fig. 5 presents a raw input image, illustrating part selection and crucial feature identification. Multiclassification predicts a category label with the highest probability for the image.

E. SYSTEM IMPLEMENTATION

In this subsection, we explain using a high-resolution smartphone camera to identify and classify bird information [40] based on deep learning. To complete the semantic bird search task, we established a client–server architecture to bridge the communication gap between the cloud and mobile device over a network. The entire setup was executed in the following manner:

- Raw bird images were distilled to remove irrelevant parts and learned by the CNN to yield parameters on the GPU platform. Subsequently, a TF inference model [41] was developed in the workstation for deployment in the smartphone.
- The output was detected using an Android app platform or through the web.

On the workstation/server side, the following segments were considered. The TF backend session model for object detection was prepared to save the TF computation graphs of input, output, weight, and bias as graph_def text files (tfdroid.pbtxt), which comprised the entire architecture of the model. The CNN architecture was trained to load the raw input data of bird images using Keras [42] callbacks with the predefined parameters into TF format to fit the model for inference. After training the model, the parameters of all saved session events of model progress in each epoch were saved as a TF checkpoint (.ckpt) file. To deploy the trained model on a smartphone, the graphs were frozen in TF format using Python. Before the trained model was frozen, a saver object was created for the session, and the checkpoints, model name, model path, and input–output parameter layers of the model were defined. All other explicit metadata assignments that were not necessary for the client–server inference, such as GPU directories on the graph nodes or graph paths, were removed. In this bird detection model, the output layer provides: (a) the parts of the input image containing a bird, (b) type of bird species, and (c) parts of the input image not containing a bird. Finally, the trained model was frozen by converting all variable parameters in the checkpoint file into constants (stops). Subsequently, both files were serialized into a single file as a ProtoBuf graph_def. The graph frozen as a ProtoBuf graph_def can be optimized, if required, for feasibility inference. The saved ProtoBuf graph_def was reloaded and resaved to a serialized string value. The following actions were considered when optimizing for inference:

- Removal of redundant variables
- Stripping out of unused nodes
- Compression of multiple nodes and expressions into a distinct node
- Removal of debug operations, such as CheckNumerics, that are not necessary for inference
- Group batch norm operation into precalculated weights
- Fusing of common operations into unified versions
- Reduction of model size through quantization and weight rounding
- Fixing of hardware assignment problems

Once the model was trained and saved for mobile inference in the workstation, we created an Android app to copy and configure the TF inference files. On the client/mobile side, the SDK written in Java and NDK written in C++ were downloaded to create mobile interface activities and to communicate with the pretrained CNN TF ProtoBuf files that contained the model definition parameters and weights. The JNI was used to bridge the TF and Android platforms. The JNI executes the loadModel function and obtains predictions of an object from the TF ProtoBuf files using the Android NDK. After classifying the object in the pertained model, the classified label output is sent back to the mobile phone using the Android NDK.

Using the aforementioned client–server computing setup, we provided a mechanism to encapsulate the cloud and mobile session. Bird recognition can be executed through cloud- and device-based inference. In this approach to deep learning inference on a mobile device, the trained model parameters are loaded into the mobile app, and the computations are completed locally on the device to predict the image output. The mobile phone is constrained by memory size and inflexibility when updating the trained model. However, in the cloud-based deep learning model, the trained model is stored on a remote server, and the server connects to the mobile device via the Internet using a web API to predict the uploaded images. Therefore, deploying the learned architecture with the cloud-based model can be easily ported to various platforms or mobile phones, and can upscale the model with new features without much difficulty. Because of the aforementioned benefits, cloud-based inference was used to execute bird image recognition. Fig. 6 shows the proposed system for bird information retrieval from the trained model stored in the workstation. The server with the TF platform takes prediction requests for bird images from client mobile phones and feeds and processes in the deep learning trained model the images sent from the API. After an image has been predicted, the TF platform classifies and generates the probability distribution of the image and transmits the query image result back to the user's mobile phone with the classified label.

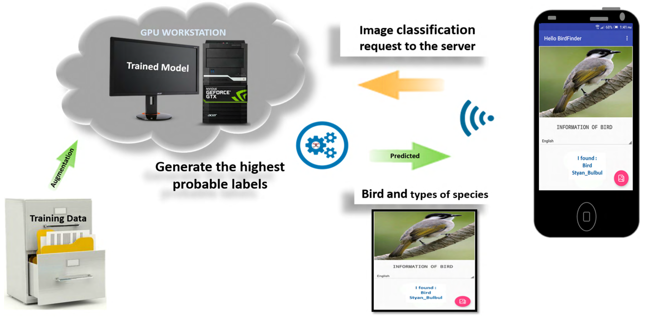
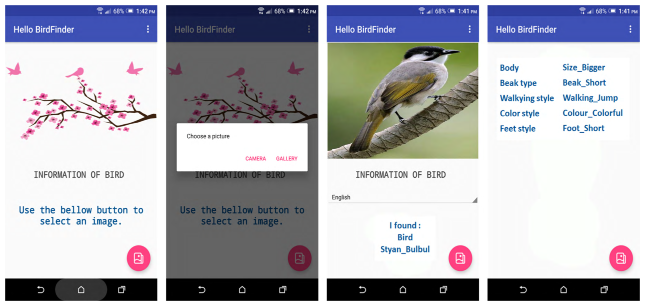

GPU WORKSTATION
Trained Model
GEFORCE GTX
Augmentation
Training Data
Generate the highest probable labels
Image classification request to the server
Predicted
Bird and types of species
Hello BirdFinder
INFORMATION OF BIRD
English
I found : Bird Styan_Bulbul

FIGURE 6. Client–server architecture for bird detection.

To analyze the uploaded images, we used a mobile phone as a client to perform the following functions: the end-user interface captures the bird image and instantly or directly uploads the image from the gallery of the mobile phone to extract image features. The mobile app sends an HTTPS request to the web server (central computer system) to retrieve the pretrained database regarding the uploaded bird image. The server performs data aggregation and an exhaustive search using the uploaded image [43] to determine the matching parameters and retrieve information related to the images. To optimize binary segmentation of the weighted graph of the image, Grabcut semantic foreground segmentation [44] is applied for bird species categorization. The head of a bird is the main prior-fitted region of interest [45], the other parts of the bird are lower priority regions of interest. A color model is projected to filter the original image with the bounding box [46]. Subsequently, the information is classified and mapped, and the correctness of the matched image is transmitted back to the user's mobile phone. The transmitted file contains metadata related to the bird's information with the classified label indicating a bird species. Fig. 7 shows the interface steps of bird detection.

V. EXPERIMENTAL RESULTS AND ANALYSIS

Fig. 8 presents 27 bird species endemic to Taiwan. The proposed system can predict and differentiate bird and nonbird images. When nonbird images are uploaded from a user's



Hello BirdFinder

INFORMATION OF BIRD

Use the bellow button to select an image.

Hello BirdFinder

Choose a picture

CAMERA GALLERY

INFORMATION OF BIRD

Use the bellow button to select an image.

Hello BirdFinder

INFORMATION OF BIRD

English

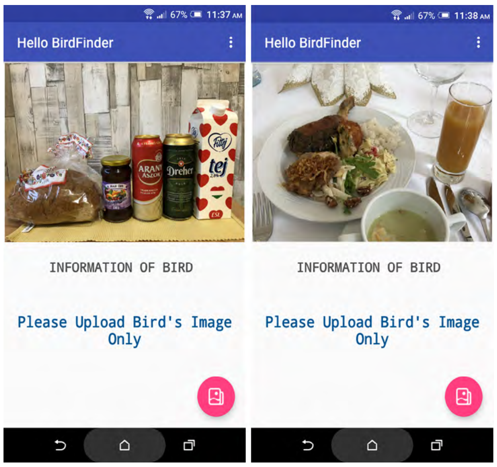
I found :
Bird
Styan_Bulbul

Hello BirdFinder

Body	Size_Bigger
Beak type	Beak_Short
Walkying style	Walking_Jump
Color style	Colour_Colorful
Feet style	Foot_Short

FIGURE 7. App interface for determining bird species.

FIGURE 8. Bird species endemic to Taiwan.



Hello BirdFinder

INFORMATION OF BIRD

Please Upload Bird's Image Only

Hello BirdFinder

INFORMATION OF BIRD

Please Upload Bird's Image Only

FIGURE 9. App interface with negative images.

smartphone, the system sends a notification to upload only bird images as shown in Fig. 9. The hardware and software specifications for inference engine execution are summarized in Table 3.

To filter nonbird images uploaded to the system automatically and to validate the effectiveness of the proposed system, 100 bird images were uploaded from a mobile phone for preliminary testing. The model achieved 100% accuracy in classifying the images as true bird pictures. Table 4 shows the bird detection results.

To acquire the output of images with or without birds, the multiscale sliding window strategy was applied so that the extracted subwindow could define the target object categories. The base learning rate was 0.01 and subsequently shifted to 0.0001. The network was trained until the cross-entropy stabilized. Skip connections were implemented when the input and output layers had equal weights. For instance, when the dimensions of the output were increased, the weights were concatenated in a deeper layer to capture and reconstruct features more effectively in the next layer. We compared different α parameters to check their influences on the final model. Table 5 compares the performances of different α values in identifying whether a bird appears in an image. In these experiments, 3563 images were split into sets of 80% for training and 20% for testing. The comparisons reveal that a high α increases the redundancy in the model; therefore, we set the α value to 0.5, which resulted in average accuracies of 100% and 99.7% for the training and test datasets, respectively.

In this study we also compared the performance of three methods such as CNN with skip connections, CNN without skip connections, and SVM with endemic bird dataset. The performance comparison of the model is set with learning rate of 0.00001 and 100 epochs. For the SVM, we used

TABLE 3. Hardware/software specifications used to execute the object detection model.

Hardware/Software	*Specification*
GPU	12 Intel Xeon CPUs, 32 GB memory, NVIDIA GeForce 2, 11GB GTX 1080Ti graphic cards.
Android Phone	5.0 or higher devices.
Android SDK, NDK	SDK is Android interface for main activity, and NDK helps to bridge the SDK and TF platform.
TensorFlow	Execute numerical computation using data flow graphs.
PyCharm	Python IDE programmers coding interface.

TABLE 4. Prediction results of images uploaded from smartphones.

Subjects	Predicted Bird	Predicted Non-bird
Bird	100%	0%
Non-bird	0%	0%

TABLE 5. Performance comparisons of different α values. T1 = training and T2 = test.

10 Fold	Accuracy α=0.3 (%)		Accuracy α=0.5 (%)		Accuracy α=0.7 (%)		Accuracy α=0.9 (%)	
	T1	T2	T1	T2	T1	T2	T1	T2
1	100.0	95.9	100.0	99.9	100.0	98.9	100.0	98.9
2	100.0	95.3	100.0	99.9	100.0	98.3	100.0	99.0
3	100.0	96.0	100.0	99.9	100.0	99.6	100.0	99.5
4	100.0	96.7	100.0	99.0	100.0	99.8	100.0	99.7
5	100.0	96.8	100.0	99.9	100.0	98.6	100.0	99.0
6	100.0	97.6	100.0	99.0	100.0	98.7	100.0	99.6
7	100.0	97.6	100.0	99.9	100.0	99.8	100.0	99.8
8	100.0	98.8	100.0	99.9	100.0	99.0	100.0	99.9
9	100.0	98.8	100.0	99.9	100.0	99.7	100.0	99.9
10	100.0	98.9	100.0	100.0	100.0	99.8	100.0	99.9
Ave.	100.0	97.2	100.0	99.7	100.0	99.2	100.0	99.5

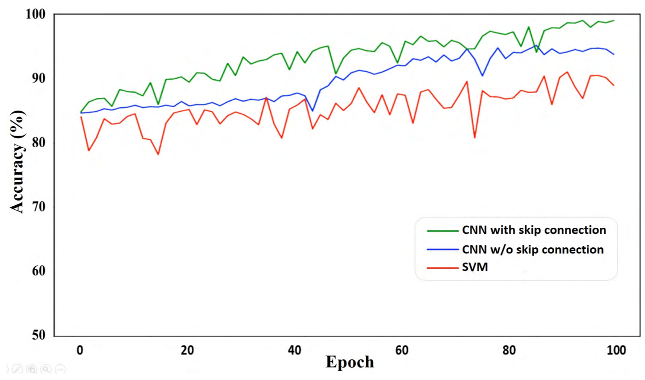


FIGURE 10. Performance comparison of the three models for the training dataset.

a linear kernel for the high dimensionality of the feature space. (with parameters gamma = 0.125 and cost (c) = 1). The models were implemented in Python 3.6.6 using the scikit-learn version 0.18.0 package. A comparison among the three models for the training dataset is shown in Fig. 10. The proposed CNN with skip connections achieved higher accuracy of 99.00% than that of 93.98% from the CNN without skip connections, and 89.00% from the SVM. This validated the effectiveness of the presented model with skip connections.

Before measuring the efficiency for the test dataset, we randomly selected numbers of images, increasing from 10 to 100 images (with an increment of 10), from the test dataset to predict their highest and five highest accuracies. If the model's top guess matched the target image, then the image was listed as Top-1. Similarly, if the model predicted the target bird at least among its top five guesses, then the image was listed as Top-5. Results from 10-fold cross-validation showed that accuracies of Top-1 were 91.20%–95.20% and of Top-5 reached 93.00%–98.50% for test bird images.

To further test the performance of the proposed system in identifying individual bird species, the following metrics were considered:

$$\text{Accuracy} = \frac{TP + TN}{TP + FP + TN + FN} \tag{3}$$

$$\text{Sensitivity} = \frac{TP}{TP + FN} \tag{4}$$

$$\text{Specificity} = \frac{TN}{TN + FP} \tag{5}$$

where TP, FP, TN, and FN represent true positive, false positive, true negative, and false negative rates, respectively. Accuracy denotes the ratio of correctly detected bird images in the entire dataset, sensitivity indicates the ratio of correctly detected birds based on the bird images, and specificity is the true negative rate of the images that are bird images. The performance evaluation of the bird images is shown in Table 6. The average sensitivity, specificity, and accuracy were 93.79%, 96.11%, and 95.37%, respectively.

TABLE 6. Performance evaluation of the classification of bird images.

Bird Images	Sensitivity (%)	Specificity (%)	Accuracy (%)
Chestnut_Bellied_Tit	91.56	95.83	94.70
Collared_Bush_Robin	91.30	94.91	94.00
Flamecrest	92.24	96.65	95.90
Gray_Cheeked_Fulvetta	95.51	97.59	97.05
Mikado_Pheasant	94.40	96.71	95.70
Rusty_Cheeked_Scimitar _Babbler	95.20	96.50	96.50
Rusty_Laughing_Thrush	92.00	94.20	93.52
Scaly_Breasted_Fulvetta	93.94	95.00	94.75
Steere_Liocichla	91.70	92.85	91.50
Streak_Breasted_Scimitar _Babbler	95.27	96.50	95.52
Streak_Throated_Fulvetta	92.23	95.62	94.45
Styan_Bulbul	90.00	95.50	95.00
Swinhoe_Pheasant	96.56	97.70	96.80
Taiwan_Bamboo_Partridge	92.35	96.63	95.74
Taiwan_Barbet	91.00	95.30	93.98
Taiwan_Barwing	96.56	97.70	96.60
Taiwan_Blue_Magpie	95.20	96.50	96.00
Taiwan_Bush_Warbler	93.94	95.00	94.85
Taiwan_Hill_Partridge	95.27	96.50	95.79
Taiwan_Hwamei	92.35	96.63	95.93
Taiwan_Rosefinch	95.27	96.50	95.65
Taiwan_Whistling_Thrush	96.56	97.70	96.80
Taiwan_Yuhina	95.27	96.50	95.95
White_Eared_Sibia	91.00	95.30	94.75
White_Throated_Laughing _Thrush	96.56	97.70	96.89
White_Whiskered Laughing _Thrush	95.27	96.50	95.75
Yellow_Tit	93.94	95.00	94.85
Ave.	93.79	96.11	95.37

VI. DISCUSSION

In this study, we developed an automatic model to classify the 27 endemic birds of Taiwan by skipped CNN model. We performed an empirical study by the skip architecture. The intuition behind using the skip connections is to provide uninterrupted gradient flow from the early layer to later layer, so that it can resolve the vanishing gradient problem. We compared the performance of various models such as CNN with skip connections, CNN without skip connections, and SVM. CNN with skip connections outperformed the other two algorithms.

However, in this study, we are more focused on predicting the 27 species of bird endemic to Taiwan more efficient and effective. The proposed model can predict the uploaded image of a bird as bird with 100% accuracy. But due to the subtle visual similarities between and among the bird species, the model sometime lacks the interspecific comparisons among the bird species and eventually leads to misclassification. In average, the test dataset yields 93.79% of sensitivity, 96.11% of specificity and this model can be used for prediction and classification of the endemic bird images.

The proposed architecture encountered some limitations and has room for improvement in the future. Sometime the model confused the prediction of endemic birds when the uploaded bird images shared similar colors and size. If most bird species within a district need to be retrieved from the system, the database must be updated and need to be retrained with new features of the birds. For extending the proposed system to some specific districts for birdwatching may encounter imbalanced distribution of the dataset among the bird species if only a small size of dataset is available.

In the future, we intend to develop a method for predicting different generations of specific bird species within the intraclass and interclass variations of birds and to expand bird species to our database so that more people can admire the beauty from watching birds.

VII. CONCLUSIONS

This study developed a mobile app platform that uses cloud-based deep learning for image processing to identify bird species from digital images uploaded by an end-user on a smartphone. This study dealt predominantly with bird recognition of 27 Taiwan endemic bird species. The proposed system could detect and differentiate uploaded images as birds with an overall accuracy of 98.70% for the training dataset. This study ultimately aimed to design an automatic system for differentiating fine-grained objects among bird images with shared fundamental characteristics but minor variations in appearance.

In the future, we intend to develop a method for predicting different generations of specific bird species within the intraclass and interclass variations of birds and to add more bird species to our database.

REFERENCES

[1] D. T. C. Cox and K. J. Gaston, "Likeability of garden birds: Importance of species knowledge & richness in connecting people to nature," *PloS one*, vol. 10, no. 11, Nov. 2015, Art. no. e0141505.

[2] 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.

[3] 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.

[4] F. Garcia, J. Cervantes, A. Lopez, and M. Alvarado, "Fruit classification by extracting color chromaticity, shape and texture features: Towards an application for supermarkets," *IEEE Latin Amer. Trans.*, vol. 14, no. 7, pp. 3434–3443, Jul. 2016.

[5] L. Zhu, J. Shen, H. Jin, L. Xie, and R. Zheng, "Landmark classification with hierarchical multi-modal exemplar feature," *IEEE Trans. Multimedia*, vol. 17, no. 7, pp. 981–993, Jul. 2015.

[6] X. Liang, L. Lin, W. Yang, P. Luo, J. Huang, and S. Yan, "Clothes co-parsing via joint image segmentation and labeling with application to clothing retrieval," *IEEE Trans. Multimedia*, vol. 18, no. 6, pp. 1175–1186, Jun. 2016.

[7] Y.-P. Huang, L. Sithole, and T.-T. Lee, "Structure from motion technique for scene detection using autonomous drone navigation," *IEEE Trans. Syst., Man, Cybern., Syst.*, to be published.

[8] C. McCool, I. Sa, F. Dayoub, C. Lehnert, T. Perez, and B. Upcroft, "Visual detection of occluded crop: For automated harvesting," in *Proc. Int. Conf. Robot. Autom. (ICRA)*, Stockholm, Sweden, May 2016, pp. 2506–2512.

[9] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "ImageNet classification with deep convolutional neural networks," in *Proc. 25th Int. Conf. Advance Neural Inf. Process. Syst.*, Lake Tahoe, NV, USA, Dec. 2012, pp. 1097–1105.

[10] B. Zhao, J. Feng, X. Wu, and S. Yan "A survey on deep learning-based fine-grained object classification and semantic segmentation," *Int. J. Automat. Comput.*, vol. 14, no. 2, pp. 119–135, Apr. 2017.

[11] H. Yang, J. T. Zhou, Y. Zhang, B.-B. Gao, J. Wu, and J. Cai, "Exploit bounding box annotations for multi-label object recognition," in *Proc. Int. Conf. Comput. Vision Pattern Recognit.*, Las Vegas, NV, USA, Jun. 2016, pp. 280–288.

[12] Li Liu, W. Ouyang, X. Wang, P. Fieguth, X. Liu, and M. Pietikäinen, "Deep learning for generic object detection: A survey," Sep. 2018, *arXiv:1809.02165*. [Online]. Available: https://arxiv.org/abs/1809.02165

[13] K. Dhindsa, K. D. Gauder, K. A. Marszalek, B. Terpou, and S. Becker, "Progressive thresholding: Shaping and specificity in automated neurofeedback training," *IEEE IEEE Trans. Neural Syst. Rehabil. Eng.*, vol. 26, no. 12, pp. 2297–2305, Dec. 2018.

[14] C.-Y. Lee, A. Bhardwaj, W. Di, V. Jagadeesh, and R. Piramuthu, "Region-based discriminative feature pooling for scene text recognition," in *Proc. Int. Conf. Comput. Vis. Pattern Recognit.*, Jun. 2014, pp. 4050–4057.

[15] B. Hariharan, P. Arbeláez, R. Girshick, and J. Malik, "Simultaneous detection and segmentation," in *Proc. Eur. Conf. Comput. Vis.*, Jul. 2014, pp. 297–312.

[16] S. Branson, G. V. Horn, S. Belongie, and P. Perona, "Bird species categorization using pose normalized deep convolutional nets," in. *Proc. Brit. Mach. Vis. Conf.*, Nottingham, U.K., Jun. 2014, pp. 1–14.

[17] B. Yao, A. Khosla, and L. Fei-Fei, "Combining randomization and discrimination for fine-grained image categorization," in *Proc. CVPR*, Colorado Springs, CO, USA, USA, Jun. 2011, pp. 1577–1584.

[18] Y.-B. Lin, Y.-W. Lin, C.-M. Huang, C.-Y. Chih, and P. Lin, "IoTtalk: A management platform for reconfigurable sensor devices," *IEEE Internet Things J.*, vol. 4, no. 5, pp. 1552–1562, Oct. 2017.

[19] 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.

[20] Google Android Developer. *Render Script API Guides*. Accessed: Nov. 20, 2018. [Online]. Available: https://developer.android.com/guide/topics/renderscript/compute.html

[21] M. Z. Andrew and G. Howard. (Jun. 14, 2017), *MobileNets: Open-Source Models for Efficient On-Device Vision*. Accessed: Feb. 20, 2018. [Online]. Available: https://research.googleblog.com/2017/06/mobilenets-open-source models-for.html

[22] C. Koylu, C. Zhao, and W. Shao, "Deep neural networks and kernel density estimation for detecting human activity patterns from Geo-tagged images: A case study of birdwatching on Flickr," *Int. J. Geo-Inf.*, vol. 8, no. 1, p. 45, Jan. 2019.

[23] R. Richa, R. Linhares, E. Comunello, A. von Wangenheim, J.-Y. Schnitzler, B. Wassmer, C. Guillemot, G. Thuret, P. Gain, G. Hager, and R. Taylor, "Fundus image mosaicking for information augmentation in computer-assisted slit-lamp imaging," *IEEE Trans. Med. Imag.*, vol. 33, no. 6, pp. 1304–1312, Jun. 2014.

[24] F. Tu, S. Yin, P. Ouyang, S. Tang, L. Liu, and S. Wei, "Deep convolutional neural network architecture with reconfigurable computation patterns," *IEEE Trans. Very Large Scale Integr. (VLSI) Syst.*, vol. 25, no. 8, pp. 2220–2233, Aug. 2017.

[25] P. Wang, W. Li, Z. Gao, J. Zhang, C. Tang, and P. O. Ogunbona, "Action recognition from depth maps using deep convolutional neural networks," *IEEE Trans. Human–Mach. Syst.*, vol. 46, no. 4, pp. 498–509, Aug. 2016.

[26] B. Du, W. Xiong, J. Wu, L. Zhang, L. Zhang, and D. Tao, "Stacked convolutional denoising auto-encoders for feature representation," *IEEE Trans. Cybern.*, vol. 47, no. 4, pp. 1017–1027, Apr. 2017.

[27] L. Yang, A. M. MacEachren, P. Mitra, and T. Onorati, "Visually-enabled active deep learning for (Geo) text and image classification: A review," *Int. J. Geo-Inf.*, vol. 7, no. 2, p. 65, Feb. 2018.

[28] J. Dean, G. Corrado, R. Monga, K. Chen, M. Devin, M. Mao, M. Ranzato, A. Senior, P. Tucker, K. Yang, Q. V. Le, and A. Y. Ng, "Large scale distributed deep networks," in *Proc. 25th Int. Conf. Adv. Neural Inf. Process. Syst.*, Lake Tahoe, NV, USA, Dec. 2012, pp. 1223–1231.

[29] J. Yosinski, J. Clune, Y. Bengio, and H. Lipson, "How transferable are features in deep neural networks," in *Proc. Int. Conf. Advance Neural Inf. Process. Syst.*, Dec. 2014, pp. 3320–3328.

[30] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *Proc. Int. Conf. Comput. Vis. Pattern Recognit.*, Jun. 2016, pp. 770–778.

[31] Nvidia. *GEFORCE GTX 1080Ti*. Accessed: Jan. 25, 2018. [Online]. Available: https://www.nvidia.com/en-us/geforce/products/10series/geforce-gtx-1080-ti/

[32] Intel. *Intel Xeon Phi Product Family*. Accessed: Jan. 2, 2018. [Online]. Available: https://www.intel.com/content/www/us/en/products/processors/xeon-phi.html?cid=sem43700027892951748&intel_term=intel+xeon&gclid=CjwKCAiAxuTQBRBmEiwAAkFF1ohAPPZlb3pEhujFDN_w9cgzqp4lPeGrui6WbsXSyW3rApIspzkhKhoCbu4QAvD_BwE&gclsrc=aw.ds

[33] Nvidia. *GPUs are Driving Energy Efficiency Across the Computing Industry, From Phones to Supercomputers*. Accessed: Nov. 25, 2018. [Online]. Available: http://www.nvidia.com/object/gcr-energy-efficiency.html

[34] H. Yao, D. Zhang, J. Li, J. Zhou, S. Zhang, and Y. Zhang, "DSP: Discriminative spatial part modeling for fine-grained visual categorization," *Image Vis. Comput.*, vol. 63, pp. 24–37, Jul. 2017.

[35] Stanford. *CS231n.2017: Convolutional Neural Networks for Visual Recognition*. Accessed: Oct. 25, 2017. [Online]. Available: http://cs231n.github.io/convolutional-networks/

[36] N. Zhang, J. Donahue, R. Girshick, and T. Darrell, "Part-based R-CNNs for fine-grained category detection," in *Proc. Int. Conf. Eur. Conf. Comput. Vis.*, Cham, Switzerland, Jul. 2014, pp. 834–849.

[37] L. Xie, J. Wang, B. Zhang, and Q. Tian, "Fine-grained image search," *IEEE Trans. Multimedia*, vol. 17, no. 5, pp. 636–647, May 2015.

[38] C. Huang, Z. He, G. Cao, and W. Cao, "Task-driven progressive part localization for fine-grained object recognition," *IEEE Trans. Multimedia*, vol. 18, no. 12, pp. 2372–2383, Dec. 2016.

[39] D. Lin, X. Shen, C. Lu, and J. Jia, "Deep LAC: Deep localization, alignment and classification for fine-grained recognition," in *Proc. Int. Conf. Comput. Vis. Pattern Recognit.*, Boston, MA, USA, Jun. 2015, pp. 1666–1674.

[40] Y.-P. Huang and T. Tsai, "A fuzzy semantic approach to retrieving bird information using handheld devices," *IEEE Intell. Syst.*, vol. 20, no. 1, pp. 16–23, Jan./Feb. 2005.

[41] TensorFlow. *Building TensorFlow on Android*. Accessed: Sep. 20, 2017. [Online]. Available: https://www.tensorflow.org/mobile/android_build

[42] Keras, *Keras: The Python Deep Learning library*. Accessed: Sep. 25, 2017. [Online]. Available: https://keras.io/

[43] J. R. R. Uijlings, K. E. A. van de Sande, T. Gevers, and A. W. M. Smeulders, "Selective search for object recognition," *Int. J. Comput. Vis.*, vol. 104, no. 2, pp. 154–171, Apr. 2013.

[44] H. Zheng, Y. Huang, H. Ling, Q. Zou, and H. Yang, "Accurate segmentation for infrared flying bird tracking," *Chin. J. Electron.*, vol. 25, no. 4, pp. 625–631, Jul. 2016.

[45] Y. Wei, W. Xia, M. Lin, J. Huang, B. Ni, J. Dong, Y. Zhao, and S. Yan, "HCP: A flexible CNN framework for multi-label image classification," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 38, no. 9, pp. 1901–1907, Sep. 2016.

[46] K.-J. Hsu, Y.-Y. Lin, and Y.-Y. Chuang, "Augmented multiple instance regression for inferring object contours in bounding boxes," *IEEE Trans. Image Process.*, vol. 23, no. 4, pp. 1722–1736, Apr. 2014.

YO-PING HUANG (S'88–M'92–SM'04) received the Ph.D. degree in electrical engineering from Texas Tech University, Lubbock, TX, USA.

He was a Professor and the Dean of Research and Development (2005–2007), the Dean of the College of Electrical Engineering and Computer Science (2002–2005), and the Department Chair (2000–2002) of Tatung University, Taipei. He is currently a Professor with the Department of Electrical Engineering, National Taipei University of Technology, Taipei, Taiwan, where he has served as the Secretary-General (2008–2011). His current research interests include fuzzy systems design and modeling, deep learning modeling, intelligent control, medical data mining, and rehabilitation systems design. He is an IET Fellow (2008) and an International Association of Grey System and Uncertain Analysis Fellow (2016). He serves as the President of the Taiwan Association of Systems Science and Engineering, the IEEE SMCS BoG, the Chair of the IEEE SMCS Technical Committee on Intelligent Transportation Systems, and the Chair of the Taiwan SIGSPATIAL ACM Chapter. He was the Chair of the IEEE SMCS Taipei Chapter, the Chair of the IEEE CIS Taipei Chapter, and the CEO of the Joint Commission of Technological and Vocational College Admission Committee, Taiwan (2011–2015).

HAOBIJAM BASANTA received the M.C.A. degree from the University of Jamia Millia Islamia, New Delhi, India. He is currently pursuing the Ph.D. degree in electrical engineering and computer science with the National Taipei University of Technology, Taipei, Taiwan. His current research interests include the Internet of Things (IoT) for the elderly healthcare systems, big data analytics, deep learning, and image processing.