Animal Species Identifier

A Project Report Submitted to

Rajiv Gandhi Proudyogiki Vishwavidyalaya, Bhopal
Towards Partial Fulfillment for the Award of

Bachelor of Technology
(Computer Science and Engineering)

Under the Supervision of
Dr. Priyanka Jangde

Submitted By
Aashish Pagare(0827CS201005)
Aditya Sharma(0827CS201016)
Aman Verma(0827CS201028)
Amit Kumar Yadav(0827CS201031)

Department of Computer Science and Engineering
Acropolis Institute of Technology & Research, Indore
Jan-June 2023

EXAMINER APPROVAL

The Project entitled " Ecommerce For Artisans" submitted by **Aashish Pagare(0827CS201005), Aditya Sharma (0827CS201016), Aman Verma(0827CS201028) and Amit Kumar Yadav (0827CS201031)** has been examined and is hereby approved towards partial fulfillment for the award of Bachelor of Technology degree in **Computer Science & Engineering** discipline, for which it has been submitted. It understood that by this approval the undersigned do not necessarily endorse or approve any statement made, opinion expressed or conclusion drawn therein, but approve the project only for the purpose for which it has been submitted.

(Internal Examiner) **(External Examiner)**
Date: **Date:**

DECLARATION
GUIDE RECOMMENDATION

This is to certify that the work embodied in this project entitled "**Animal Species Identifier** " submitted by **Aashish Pagare(0827CS201005), Aditya Sharma (0827CS201016), Aman Verma(0827CS201028) and Amit Kumar Yadav (0827CS201031)** is a satisfactory account of the bonafide work done under the supervision of **Prof. Priyanka Jangde** are recommended towards partial fulfillment for the award of the Bachelor of Technology (**Computer Science & Engineering**) degree by Rajiv Gandhi Proudyogiki Vishwavidhyalaya, Bhopal.

(Project Guide) **(Project Coordinator)**

ACKNOWLEDGEMENT

We thank the almighty Lord for giving me the strength and courage to sail out through the tough and reach on shore safely. There are a number of people without whom this project's work would not have been feasible. Their high academic standards and personal integrity provided me with continuous guidance and support. We owe a debt of sincere gratitude, deep sense of reverence and respect to our guide and mentors **Prof. Priyanka Jangde**, Associate Professor, AITR, for their motivation, sagacious guidance, constant encouragement, vigilant supervision and valuable critical appreciation throughout this project work, which helped us to successfully complete the project on time. We express profound gratitude and heartfelt thanks to **Dr Kamal Kumar Sethi**, HOD CSE, AITR Indore for his support, suggestion and inspiration for carrying out this project. I am very much thankful to other faculty and staff members of CSE Dept, AITR Indore for providing me all support, help and advice during the project. We would be failing in our duty if we do not acknowledge the support and guidance received from **Dr S C Sharma**, Director, AITR, Indore whenever needed. We take the opportunity to convey my regards to the management of Acropolis Institute, Indore for extending academic and administrative support and providing me with all necessary facilities for the project to achieve our objectives. We are grateful to our parents and family members who have always loved and supported us unconditionally. To all of them, we want to say, “Thank you”, for being the best family that one could ever have and without whom none of this would have been possible.

Aashish Pagare (0827CS201031)

Aditya Sharma (0827CS201016)

Aman Verma (0827CS201028)

Amit Kumar Yadav (0827CS201031)

EXECUTIVE SUMMARY

This project is submitted to Rajiv Gandhi Proudyogiki Vishwavidhyalaya, Bhopal(MP), India for partial fulfillment of Bachelor of Technology in **Computer Science & Engineering** branch under the sagacious guidance and vigilant supervision of **Prof. Priyanka Jangde.** The project is a Animal Species Identifier. In the project, Deep Learning is used . This project is mainly for the artisans. Artisans can sell their artwork through this site and consumers can buy it .

Art is an Irreplaceable Way Of Understanding And Expressing the World

-Dana Gioia

CONTENTS

TITLE **Page No.**

LIST OF FIGURES

LIST OF TABLES

1.INTRODUCTION

1.1. Introduction:

Bird behaviour and populace patterns have become a significant issue now a days. Birds help us to recognize different life forms on the earth effectively as they react rapidly to ecological changes. Be that as it may, assembling and gathering data about bird species requires immense human exertion just as it turns into an extremely costly technique. In such a case, a solid framework that will give enormous scale preparation of data about birds and will fill in as a significant apparatus for scientists, legislative offices, and so forth is required. In this way, bird species distinguishing proof assumes a significant job in recognizing that a specific picture of birds has a place with which categories. Bird species identification means predicting the bird species belongs to which category by using an image.

The recognition of bird species can be possible through a picture, audio or video. An audio processing method makes it conceivable to recognize by catching the sound sign of different birds. Be that as it may, because of the blended sounds in condition, for example, creepy crawlies, objects from the real world, and so forth handling of such data turns out to be progressively convoluted. Normally, people discover images more effectively than sounds or recordings. So, an approach to classify birds using an image over audio or video is preferred. Bird species identification is a challenging task to humans as well as to computational procedures that carry out such a task in an automated fashion.

As image-based classification systems are improving the task of classifying, objects are moving into datasets with far more categories such as Caltech-UCSD. Recent work has seen much success in this area. Caltech UCSD Birds 200(CUB-200-2011) is a well-known dataset for bird images with photos of 200 categories. The dataset contains birds that are mostly found in Northern America. Caltech-UCSD Birds 200 consists of 11,788 images and annotations like 15 Part Locations, 312 Binary Attributes, 1 Bounding Box.

1.1.1. Machine Learning

Machine Learning is the most popular technique of predicting or classifying information to help people in making necessary decisions. Machine Learning algorithms are trained over instances or examples through which they learn from past experiences and analyse the historical data.

Simply building models is not enough. You must also optimize and tune the model appropriately so that it provides you with accurate results. Optimization techniques involve tuning the hyperparameters to reach an optimum result.

As it trains over the examples, again and again, it is able to identify patterns in order to make decisions more accurately. Whenever any new input is introduced to the ML model, it applies its learned patterns over the new data to make future predictions. Based on the final accuracy, one can optimize their models using various standardized approaches. In this way, Machine Learning model learns to adapt to new examples and produce better results.

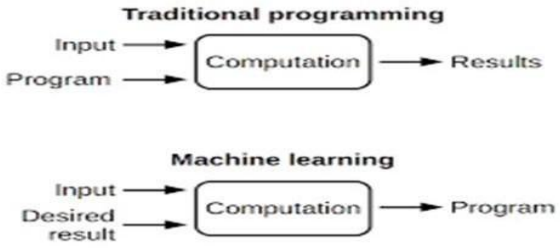


Fig 1. Machine Learning vs Traditional Programming

Types of Learnings

Machine Learning Algorithms can be classified into 3 types as follows:

1. Supervised learning
2. Unsupervised Learning
3. Reinforcement Learning

1.1.2. Supervised Learning

Supervised learning is the most popular paradigm for machine learning. It is the easiest to understand and the simplest to implement. It is the task of learning a function that maps an input to an output based on example input-output pairs. It infers a function from labelled training data consisting of a set of training examples. In supervised learning, each example is a pair consisting of an input object (typically a vector) and a desired output value (also called the supervisory signal). A supervised learning algorithm analyses the training data and produces an inferred function, which can be used for mapping new examples. Supervised Learning is very similar to teaching a child with the given data and that data is in the form of examples with labels, we can feed a learning algorithm with these example-label pairs one by one, allowing the algorithm to predict the right answer or not. Over time, the algorithm will learn to approximate the exact nature of the relationship between examples and their labels. When fully trained, the supervised learning algorithm will be able to observe a new, never-before-seen example and predict a good label for it.

Most of the practical machine learning uses supervised learning. Supervised learning is where you have input variable (x) and an output variable (Y) and you use an algorithm to learn the mapping function from the input to the output.

$$Y = f(x)$$

The goal is to approximate the mapping function so well that when you have new input data (x) that you can predict the output variables (Y) for the data. It is called supervised learning because the process of an algorithm learning from the training dataset can be thought of as a teacher supervising the learning process. Supervised learning is often described as task oriented. It is highly focused on a singular task, feeding more and more examples to the algorithm until it can accurately perform on

that task.

This is the learning type that you will most likely encounter, as it is exhibited in many of the common applications like **Advertisement Popularity, Spam Classification, face recognition.**

Two types of Supervised Learning are:

1. **Regression:**

   Regression models a target prediction value based on independent variables. It is mostly used for finding out the **relationship between variables** and **forecasting**. Regression can be used to estimate/ predict **continuous values** (Real valued output). For example**,** given a picture of a person then we have to predict the age on the basis of the given picture.

2. **Classification:**

   Classification means to **group** the output into a class. If the data is **discrete** or **categorical** then it is a classification problem. For example**,** given data about the sizes of houses in the real estate market, making our output about whether the house "sells for **more** or **less** than the asking price" i.e. Classifying houses into two discrete categories.

1.1.3. Unsupervised Learning

Unsupervised Learning is a machine learning technique, where you do not need to supervise the model. Instead, you need to allow the model to work on its own to discover information. It mainly deals with the unlabelled data and looks for previously undetected patterns in a data set with no pre-existing labels and with a minimum of human supervision. In contrast to supervised learning that usually makes use of human- labelled data, unsupervised learning, also known as self-organization, allows for modelling of probability densities over inputs.

Unsupervised machine learning algorithms infer patterns from a dataset without reference to known, or labelled outcomes. It is the training of machine using information that is neither classified nor labelled and allowing the algorithm to act on that information without guidance. Here the task of machine is to group unsorted information according to similarities, patterns, and differences without any prior training of data. Unlike supervised learning, no teacher is provided that means no

training will be given

to the machine. Therefore, machine is restricted to find the hidden structure in unlabelled data by our-self. For example, if we provide some pictures of dogs and cats to the machine to categorized, then initially the machine has no idea about the features of dogs and cats so it categorize them according to their similarities, patterns and differences. The Unsupervised Learning algorithms allows you to perform more complex processing tasks compared to supervised learning. Although, unsupervised learning can be more unpredictable compared with other natural learning methods.

Unsupervised learning problems are classified into two categories of algorithms:

- **Clustering**: A clustering problem is where you want to discover the inherent groupings in the data, such as grouping customers by purchasing behaviour.
- **Association**: An association rule learning problem is where you want to discover rules that describe large portions of your data, such as people that buy X also tend to buy Y.

1.1.4. Reinforcement Learning

Reinforcement Learning (RL) is a type of machine learning technique that enables an agent to learn in an interactive environment by trial and error using feedback from its own actions and experiences. Machine mainly learns from past experiences and tries to perform best possible solution to a certain problem. It is the training of machine learning models to make a sequence of decisions. Though both supervised and reinforcement learning use mapping between input and output, unlike supervised learning where the feedback provided to the agent is **correct set of actions** for performing a task, reinforcement learning uses **rewards and punishments** as signals for positive and negative behaviour. Reinforcement learning is currently the most effective way to hint machine's creativity.

1.1.5. Neural Networks

Neural Network (or Artificial Neural Network) has the ability to learn by examples. ANN is an information processing model inspired by the biological neuron system. ANN biologically inspired simulations that are performed on the computer to do a certain specific set of tasks like clustering, classification, pattern recognition etc. It is composed of a large number of highly interconnected processing elements known as the neuron to solve problems. It follows the non-linear path and process information in parallel throughout the nodes. A neural network is a complex adaptive system. Adaptive means it has the ability to change its internal structure by adjusting weights of inputs.

Artificial Neural Networks can be best viewed as weighted directed graphs, where the nodes are formed by the artificial neurons and the connection between the neuron outputs and neuron inputs can be represented by the directed edges with weights. The ANN receives the input signal from the external world in the form of a pattern and image in the form of a vector. These inputs are then mathematically designated by the notations x(n) for every n number of inputs. Each of the input is then multiplied by its corresponding weights (these weights are the details used by the artificial neural networks to solve a certain problem). These weights typically represent the strength of the interconnection amongst neurons inside the artificial neural network. All the weighted inputs are summed up inside the computing unit (yet another artificial neuron).

If the weighted sum equates to zero, a bias is added to make the output non-zero or else to scale up to the system's response. Bias has the weight and the input to it is always equal to 1. Here the sum of weighted inputs can be in the range of 0 to positive infinity. To keep the response in the limits of the desired values, a certain threshold value is benchmarked. And then the sum of weighted inputs is passed through the activation function. The activation function is the set of transfer functions used to get the desired output of it. There are various flavours of the activation function, but mainly either linear or non-linear set of functions. Some of the most commonly used set of activation functions are the Binary, Sigmoid (linear) and Tan hyperbolic sigmoidal (non-linear) activation functions.

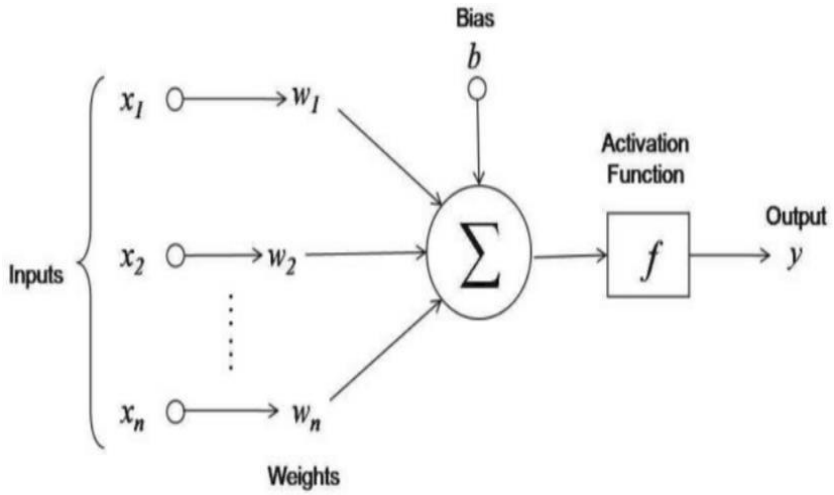


Fig 2. Neuron Model

The Artificial Neural Network contains three layers

1. **Input Layer:** The input layers contain those artificial neurons (termed as units) which are to receive input from the outside world. This is where the actual learning on the network happens or corresponding happens else it will process.
2. **Hidden Layer:** The hidden layers are mentioned hidden in between input and the output layers. The only job of a hidden layer is to transform the input into something meaningful that the output layer/unit can use in some way. Most of the artificial neural networks are all interconnected, which means that each of the hidden layers is individually connected to the neurons in its input layer and also to its output layer leaving nothing to hang in the air. This makes it possible for a complete learning process and also learning occurs to the maximum when the weights inside the artificial neural network get updated after each iteration.
3. **Output Layer:** The output layers contain units that respond to the information that is fed into the system and also whether it learned any task or not.

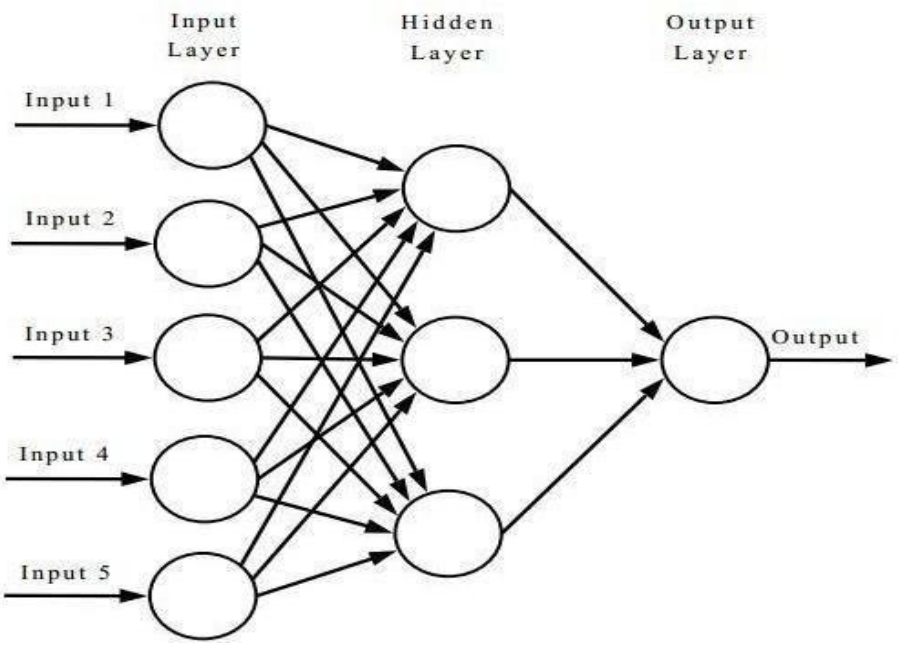


Fig 3. Basic Neural Network

Learning process of a neural network:

1. Start with values (often random) for the network parameters (*wij* weights and *bj* biases).

2. Take a set of examples of input data and pass them through the network to obtain their prediction.

3. Compare these predictions obtained with the values of expected labels and calculate the loss with them.

4. Perform the backpropagation in order to propagate this loss to each and every one of the parameters that make up the model of the neural network.

5. Use this propagated information to update the parameters of the neural network with the gradient descent in a way that the total loss is reduced, and a better model is obtained.

6. Continue iterating in the previous steps until we consider that we have a good model.

1.1.6 Deep Learning

Deep learning is a branch of machine learning which is completely based on artificial neural networks. Deep learning is an artificial intelligence function that imitates the workings of the human brain in processing data and creating patterns for use in decision making. Deep learning is a subset of machine learning in artificial intelligence (AI) that has networks capable of learning unsupervised from data that is unstructured or unlabelled. It has a greater number of hidden layers and known as deep neural learning or deep neural network. Deep learning has evolved hand-in-hand with the digital era, which has brought about an explosion of data in all forms and from every region of the world. This data, known simply as big data, is drawn from sources like social media, internet search engines, e-commerce platforms, and online cinemas, among others. This enormous amount of data is readily accessible and can be shared through fintech applications like cloud computing. However, the data, which normally is unstructured, is so vast that it could take decades for humans to comprehend it and extract relevant information. Companies realize the incredible potential that can result from unravelling this wealth of information and are increasingly adapting to AI systems for automated support. Deep learning learns from vast amounts of unstructured data that would normally take humans decades to understand and process. Deep learning and utilizes a hierarchical level of artificial neural networks to carry out the process of machine learning. The artificial neural networks are built like the human brain, with neuron nodes connected like a web. While traditional programs build analysis with data in a linear way, the hierarchical function of deep learning systems enables machines to process data with a nonlinear approach.

Deep Neural Network

It is a neural network with a certain level of complexity (having multiple hidden layers in between input and output layers). They are capable of modelling and processing non-linear relationships.

Disadvantages of previous bird classification through voice:

- Background noise- especially while using data recorded in a city.
- Multi-label classification problem-when there are many species singing at the same time.
- Different types of bird songs.
- Inter-species variance-there might be difference in bird song between the same species living in different regions or countries.
- Data set issues-the data can be highly imbalanced due to bigger popularity of one species over another, there is a large number of different species and recordings can have different length, quality of recordings.

Objective of the project:

- The main objective of the project is to identify the species of the birds by analysing an image. Some of the experts like ornithologists couldn't identify species of the bird correctly by looking at an image.
- Although bird classification can be done manually by domain experts, with growing amounts of data, this rapidly becomes a tedious and time-consuming process. So, by this model we can identify the species of the birds accurately and in less time.

1.2. Problem Statement:

Identifying a bird can be a challenge, even for experienced birders. And if you're new to using field guides, it can be daunting to figure out where to even begin searching in the hundreds of pages of species. By some features like size, shape and colour birds can be classified. By using CNN, we can classify the species of the birds.

2. LITERATURE SURVEY

2.1. Introduction:

In particular, it was discovered that ecologists monitor them to determine the factors causing population fluctuation and to help in conserving and managing threatened and endangered species. The various surveys used in counting bird species including data collection techniques were succinctly reviewed. It was established that a small but growing number of researchers have studied the use of computer vision for monitoring species of birds.

This chapter evaluates reports of studies found in literature that are related to monitoring and classification of species. In particular, it focuses on reviewing bird techniques used for these species are often similar, and motion features which this research seeks to investigate for classification of birds. First techniques which perform classification using single images were explored. This was done by reviewing them separately as those that are used for classification of bird species and those for other species specifically bats.

2.2. Survey of different classification methods:

John Martinsson et al (2017) [1], presented the CNN algorithm and deep residual neural networks to detect an image in two ways i.e., based on feature extraction and signal classification. They did an experimental analysis for datasets consisting of different images. But their work didn't consider the background species. In Order to identify the background species larger volumes of training data are required, which may not be available.

Juha Niemi, Juha T Tanttu et al (2018) [2], proposed a Convolutional neural network trained with deep learning algorithms for image classification. It also proposed a data augmentation method in which images are converted and rotated in accordance with the desired color. The final identification is based on a fusion of parameters provided by the radar and predictions of the image classifier.

Li Jian, Zhang Lei et al (2014)[3], proposed an effective automatic bird species identification based on the analysis of image features. Used the database of standard images and the algorithm of similarity comparisons.

Madhuri A. Tayal, Atharva Magrulkar et al (2018)[4], developed a software application that is used to simplify the bird identification process. This bird identification software takes an image as an input and gives the identity of the bird as an output. The technology used is transfer learning and MATLAB for the identification process.

Andreia Marini, Jacques Facon et al (2013) [5], proposed a novel approach based on color features extracted from unconstrained images, applying a color segmentation algorithm in an attempt to eliminate background elements and to delimit candidate regions where the bird may be present within the image. Aggregation processing was employed to reduce the number of intervals of the histograms to a fixed number of bins. In this paper, the authors experimented with the CUB-200 dataset and results show that this technique is more accurate.

Marcelo T. Lopes, Lucas L. Gioppo et al (2011) [6], focused on the automatic identification of bird species from their audio recorded song. Here the authors dealt with the bird species identification problem using signal processing and machine learning techniques with the MARSYAS feature set. Presented a series of experiments conducted in a database composed of bird songs from 75 species out of which problem obtained in performance with 12 species.

Peter Jancovic and Munevver Kokuer et al (2012) [7], investigated acoustic modelling for recognition of bird species from audio field recordings. Developed a hybrid deep neural network hidden Markov model (DNN-HMM). The developed models were employed for bird species identification, detection of specific species and recognition of multiple bird species vocalizing in a given recording. In this paper, the authors achieved an identification accuracy of 98.7% and recognition accuracy of 97.3%.

Mario Lasseck et al (2013) [8], presented deep convolutional neural networks and data augmentation techniques for audio-based bird species identification. In this paper, the author used the Xeno-Canto set of audio recordings of bird species.

2.3. Existing System:

To identify the bird species there are many websites produces the results using different technologies. But the results are not accurate. For suppose if we will give an input in those websites and android applications it gives us multiple results instead of single bird name. It shows us the all bird names which are having similar characteristics. So, we aimed to develop a project to produce better and accurate results. In order to achieve this, we have used Convolutional Neural Networks to classify the bird species.

3. METHODOLOGY

3.1. Proposed System:

Convolution neural network algorithm is a multilayer perceptron that is the special design for the identification of two-dimensional image information. It has four layers: an input layer, a convolution layer, a sample layer, and an output layer. In a deep network architecture, the convolution layer and sample layer may have multiple. CNN is not as restricted as the Boltzmann machine, it needs to be before and after the layer of neurons in the adjacent layer for all connections, convolution neural network algorithms, each neuron doesn't need to experience the global image, just feel the local region of the image. In addition, each neuron parameter is set to the same, namely, the sharing of weights, namely each neuron with the same convolution kernels to the deconvolution image.

The key era of CNN is the local receptive field, sharing of weights, subsampling by using time or space, with a purpose to extract features and reduce the size of the training parameters. The advantage of CNN algorithm is to avoid the explicit feature extraction, and implicitly to learn from the training data. The same neuron weights on the surface of the feature mapping, thus the network can learn parallel, and reduce the complexity of the network Adopting sub-sampling structure by time robustness, scale, and deformation displacement. Input information and network topology can be a very good match. It has unique advantages in image processing. The Convolution Neural Network involves these steps

Convolution Layer:

The convolutional layer is the core constructing block of a CNN. The convolution layer comprises a set of independent feature detectors. Each Feature map is independently convolved with the images.

Pooling Layer:

The pooling layer feature is to progressively reduce the spatial size of the illustration to reduce the wide variety of parameters and computation in the network. The pooling layer operates on each function map independently. The approaches used in pooling are:

- Max Pooling
- Mean Pooling
- Sum Pooling

Why to use pooling layers:

- Pooling layers are used to reduce the dimensions of the feature maps. Thus, it reduces the number of parameters to learn and the amount of computation performed in the network.
- The pooling layer summarises the features present in a region of the feature map generated by a convolution layer. So, further operations are performed on summarised features instead of precisely positioned features generated by the convolution layer. This makes the model more robust to variations in the position of the features in the input image.

Fully Connected Layer:

Neurons in the fully connected layer have full connections to all activations inside the preceding layer. In this, the output obtained from max pooling is converted to a one- dimensional array and that should be the input layer and the process continues the same as the ANN model.

3.2. Architecture:

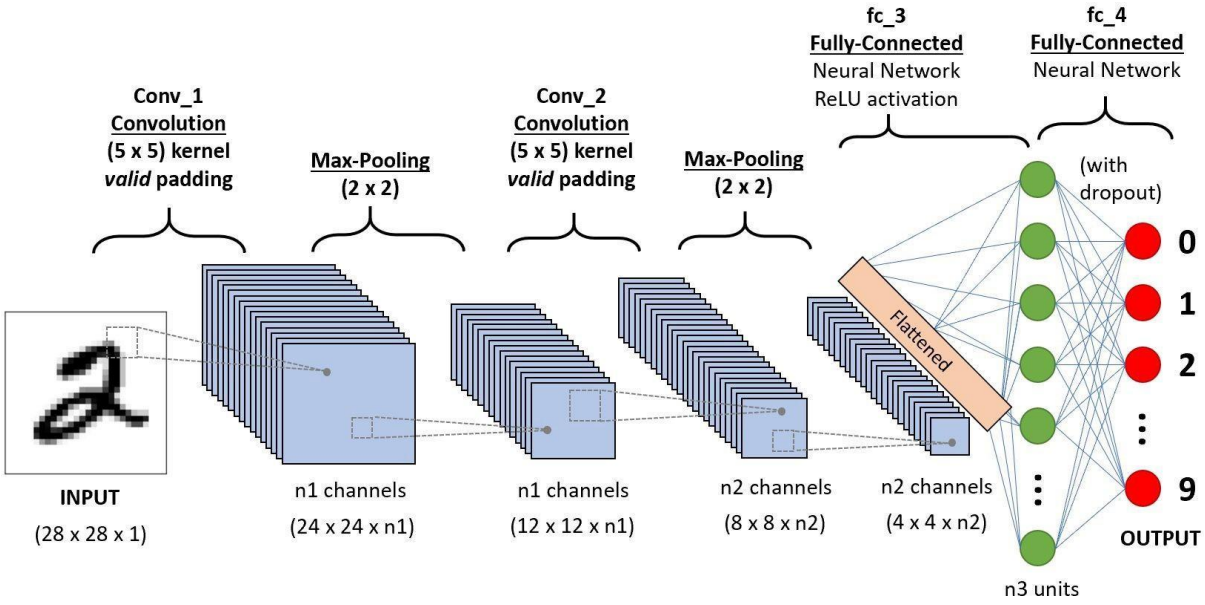


Fig4. A CNN Sequence

A Convolutional Neural Network (ConvNet/CNN) is a Deep Learning algorithm which can take in an input image, assign importance (learnable weights and biases) to various aspects/objects in the image and be able to differentiate one from the other. The pre-processing required in a ConvNet is much lower as compared to other classification algorithms. While in primitive methods filters are hand-engineered, with enough training, ConvNets have the ability to learn these filters/characteristics.

A ConvNet is able to successfully capture the Spatial and Temporal dependencies in an image through the application of relevant filters. The architecture performs a better fitting to the image dataset due to the reduction in the number of parameters involved and reusability of weights. In other words, the network can be trained to understand the sophistication of the image better.

The role of the ConvNet is to reduce the images into a form which is easier to process, without losing features which are critical for getting a good prediction.

Convolution Layer

Filters (Convolution Kernels)

A filter (or kernel) is an integral component of the layered architecture.

Generally, it refers to an operator applied to the entirety of the image such that it transforms the information encoded in the pixels. In practice, however, a kernel is a smaller-sized matrix in comparison to the input dimensions of the image, that consists of real valued entries.

The real values of the kernel matrix change with each learning iteration over the training set, indicating that the network is learning to identify which regions are of significance for extracting features from the data.

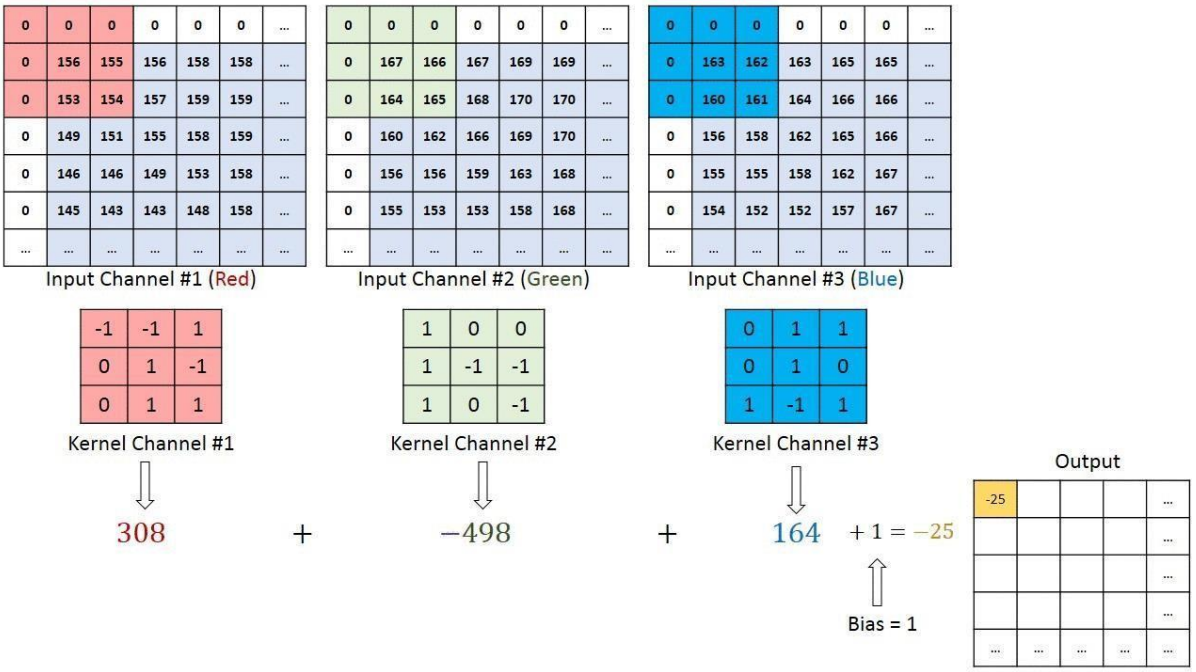


Fig 5. Convolution operation with kernel

In Fig 5 we a convoluting a 5x5x1 image with a 3x3x1 kernel (which change each iteration to extract a significant features) to get a 3x3x1 convolved feature. The filter moves to the right with a certain stride value till it parses the complete width. In case of images with multiple channels (e.g. RGB) the kernel has a same depth as that of input image. Matrix multiplication is performed between Kn and In stack ([K1, I1]; [K2, I2];

[K3, I3]) and all results are summed with bias to give us a squashed 1 depth channel convoluted feature output.

The objective of the Convolution Operation is to extract the high-level features such as edges, from the input image, the first Convolutional Layer is responsible for capturing the Low-Level features such as edges, color, gradient orientation, etc.

Kernel Size - 3x3

Pooling Matrix Size - 2x2

Image will be Resized to - (Height=250, Width=250)

Pooling Layer:

(a)Convoluted Output (b) Pooling Output

Fig 6. Performing Pooling operation

The Pooling layer is responsible for reducing the spatial size of the Convolved Feature. This is to decrease the computational power required to process the data through dimensionality reduction. Furthermore, it is useful for extracting dominant features which are rotational and positional invariant, thus maintaining the process of effectively training of the model.

There are three types of Pooling: Max Pooling, Average Pooling and Global Pooling.

Max Pooling: Max pooling is a pooling operation that selects the maximum element from the region of the feature map covered by the filter. Thus, the output after max-pooling layer would be a feature map containing the most prominent features of the previous feature map. In Fig. 3 we perform maximum pooling operation by considering convoluted feature output obtained from convolution layer.

Average Pooling: Average pooling computes the average of the elements present in the region of feature map covered by the filter. Thus, while max pooling gives the most prominent feature in a particular patch of the feature map, average pooling gives the average of features present in a patch.

Global Pooling: Global pooling reduces each channel in the feature map to a single value. Thus, an n_h x n_w x n_c feature map is reduced to 1 x 1 x n_c feature map. This is equivalent to using a filter of dimensions n_h x n_w i.e. the dimensions of the feature map. Further, it can be either global max pooling or global average pooling.

3.3. Data Set Description:

For the project, we will be using the dataset that includes 200 categories of bird species, 11,788 total number of images, and other information such as labelled visible bird parts, binary attributes, and bounding boxes surrounding the birds. The authors also include a recommended test and training set split of the data. From these 200 categories we are taking 3 basic categories and performs the test.

Fig 7. Sample images dataset

Input data for the identification system consist of digital images and parameters from the radar. All images for training the CNN are of wild birds in flight and they have been taken manually at the test location. There are also constraints concerning the area where the images have to be taken. Here, the area refers to the air space in the vicinity of the pilot wind turbine. We have used the wind turbine swept area as a suitable altitude level constraint for taking the images, because birds flying below or above the swept area are not in danger.

The bird species used in this dataset are:

- Painted_Bunting
- Breweer_Blackbird

- Eared_Grebe

1.Painted_Bunting:

The **painted bunting** (*Passerina ciris*) is a species of bird in the cardinal family, Cardinalidae, that is native to North America. The bright plumage of the male only comes in the second year of life; in the first year they can only be distinguished from the female by close inspection.

Sometimes called the "Nonpareil," meaning "unrivalled," a fair way to describe the unbelievable colors of the male Painted Bunting. This species is locally common in the Southeast, around brushy areas and woodland edges. It is often secretive, staying low in dense cover. However, males sing their bright warbling songs from higher in the trees, partly hidden among foliage or sometimes out in the sun on an exposed perch. Some lucky Floridians have Painted Buntings coming to their bird feeders in winter.

Here are few pictures of Painted_Bunting:

Fig 8 . Sample Image of Painted_Bunting

Fig 9. Sample image of Painted_Bunting

2.Brewer_Blackbird:

The **Brewer's blackbird** (*Euphagus cyanocephalus*) is a medium-sized New World blackbird. It is named after the ornithologist Thomas Mayo Brewer.

This is the common blackbird of open country in the West, often seen walking on the ground with short forward jerks of its head. It adapts well to habitats altered by humans, and in places it may walk about on suburban sidewalks or scavenge for crumbs around beachfront restaurants. In winter, Brewer's Blackbirds gather in large flocks, often with other blackbirds, and may be seen foraging in farmland all across the western and southern states.

Adult males have black plumage with an iridescent purple head and neck and glossy bluish-green highlights on the rest of the body. The feet and legs are black and the eye is bright yellow. The female is brownish-grey with slight hints of the male's iridescence. The female's eye is dark brown, while the male's is bright yellow. Overall, they resemble the eastern member of the same genus, the rusty blackbird; however, the Brewer's blackbird has a shorter bill and the male's head is iridescent purple. This bird is often mistaken for the common grackle but has a shorter tail. The call is a sharp check which is also distinguishable. This bird is in a different family from the Eurasian blackbird.

Here are few pictures of Brewer_Blackbird:

Fig 10. Sample image of Brewer_Blackbird

Fig 11. Sample Image of Brewer_Blackbird

3.Eared_Grebe:

The black-necked grebe (Podiceps nigricollis), known in North America as the **eared grebe**, is a member of the grebe family of water birds. It was described in 1831 by Christian Ludwig Brehm. There are currently three accepted subspecies, including the nominate subspecies. Its breeding plumage features a distinctive ochre-coloured plumage which extends behind its eye and over its ear coverts. The rest of the upper

parts, including the head, neck, and breast, are coloured black to blackish brown. The flanks are tawny rufous to maroon-chestnut, and the abdomen is white. When in its non- breeding plumage, this bird has greyish-black upper parts, including the top of the head and a vertical stripe on the back of the neck. The flanks are also greyish-black. The rest of the body is a white or whitish colour. The juvenile has more brown in its darker areas. The subspecies californicus can be distinguished from the nominate by the former's usually longer bill.

Here are some pictures of Eared_Grebe:

Fig 12. Sample Image of Eared_Grebe

Fig 13. Sample Image of Eared_Grebe

Dataset Details:

Sno.	Bird Name	Samples count	Dimensions of the image
1.	Painted_Bunting	250	Any Size
2.	Brewer_Blackbird	175	Any Size
3.	Eared_Grebe	200	Any Size

Table 1. Dataset Details

Here we can give any size of the image it will be resized to the dimensions 250x250.

The Features considered in this classification are : Beak ,Shape ,Neck, Color, Tail etc.,

3.4. Algorithm:

In this experiment, unsupervised learning algorithm has been used for developing the system, because the inputted image defined is not known. Also, the data which is given to unsupervised learning algorithm are not labeled, i.e. only the input variables(X) are given with no corresponding output variables. In unsupervised learning, algorithms discover interesting structures in the data themselves. In detail, clustering is used for dividing the data into several groups.

In depth, deep learning models used to find vast number of neurons. Deep learning algorithms learn more about the image as it goes through each neural network layer. For classifying **Neural Network** is used. Figure 5 represents layers of neural networks for feature extraction. The neural network is a framework for many machine learning algorithms. Neural networks consist of vector of weights(W) and the bias (B).

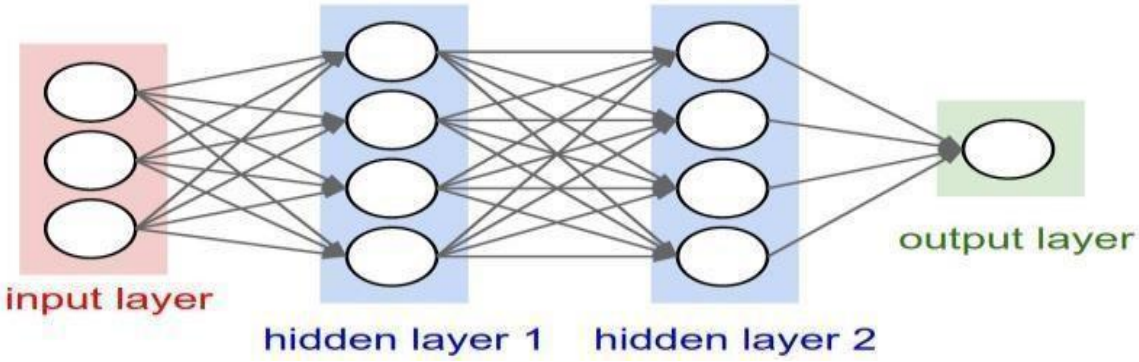


Fig 14. Three layers of Neural Network

In deep learning, convolutional neural network (CNN) is a class of deep neural network mostly used for analyzing visual images. It consists of an input layer and output layer as well as multiple hidden layers. Every layer is made up of group of neurons and each layer is fully connected to all neurons of its previous layer. The output layer is responsible for prediction of output. The convolutional layer takes an image as input, and produces a set of feature maps as output. The input image can contain multiple channels such as color, wings, eyes, beak of birds which means that the convolutional layer perform a mapping from 3D volume to another 3D volume. 3D volumes considered are width, height, depth. The CNN have two components:

1) Feature extraction part: features are detected when network performs a series of convolutional and pooling operation.
2) Classification part: extracted features are given to fully connected layer which acts as classifier.

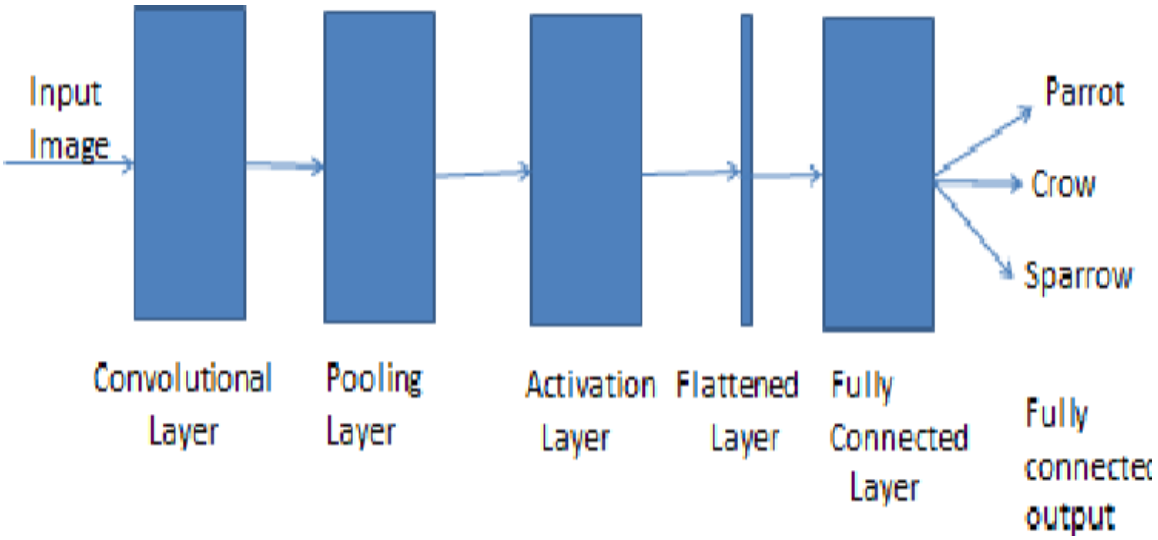


Fig 15. Convolutional Neural Network Layers

CNN consists of four layers: convolutional layer, activation layer, pooling layer and fully connected. Convolutional layer allows extracting visual features from an image in small amounts. Pooling is used to reduce the number of neurons from previous convolutional layer but maintaining the important information. Activation layer passes a value through a function which compresses values into range. Fully connected layer connects a neuron from one layer to every neuron in another layer. As CNN classifies each neuron in depth, so it provides more accuracy.

Image classification: image classification in machine learning is commonly done in two ways:

1) Gray scale
2) Using RGB values

Normally all the data is mostly converted into gray scale. In gray scale algorithm,

computer will assign values to each pixel based on how the value of the pixel is it**.** All the pixel values are put into an array and the computer will perform operation on that array to classify the data.

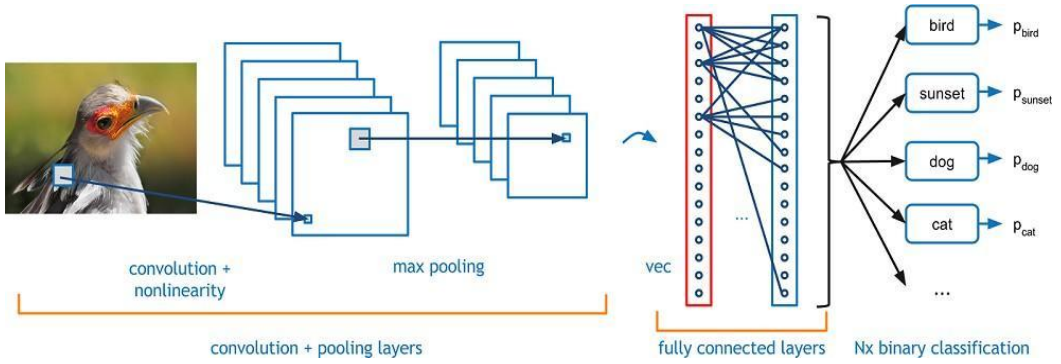


Fig 16. Understanding of CNN layers

Image classification is the task of taking an input image and outputting a class (a cat, dog, etc) or a probability of classes that best describes the image. For humans, this task of recognition is one of the first skills we learn from the moment we are born and is one that comes naturally and effortlessly as adults.

When a computer sees an image (takes an image as input), it will see an array of pixel values. Depending on the resolution and size of the image, it will see a 32 x 32 x 3 array of numbers (The 3 refers to RGB values). Just to drive home the point, let's say we have a color image in JPG form and its size is 480 x 480. The representative array will be 480 x 480 x 3. Each of these numbers is given a value from 0 to 255 which describes the pixel intensity at that point. These numbers, while meaningless to us when we perform image classification, are the only inputs available to the computer. The idea is that you give the computer this array of numbers and it will output numbers that describe the probability of the image being a certain class (.80 for cat, .15 for dog, .05 for bird, etc).

Now that we know the problem as well as the inputs and outputs, let’s think about how to approach this. What we want the computer to do is to be able to differentiate between all the images it’s given and figure out the unique features that make a dog a dog or that make a cat a cat. This is the process that goes on in our minds subconsciously as well. When we look at a picture of a dog, we can classify it as such if

the picture has identifiable features such as paws or 4 legs. In a similar way, the computer is able

perform image classification by looking for low level features such as edges and curves, and then building up to more abstract concepts through a series of convolutional layers. This is a general overview of what a CNN does.

The first layer in a CNN is always a **Convolutional Layer**. First thing to make sure you remember is what the input to this conv (I'll be using that abbreviation a lot) layer is. Like we mentioned before, the input is a 32 x 32 x 3 array of pixel values. Now, the best way to explain a conv layer is to imagine a flashlight that is shining over the top left of the image. Let's say that the light this flashlight shines covers a 5 x 5 area. And now, let's imagine this flashlight sliding across all the areas of the input image. In machine learning terms, this flashlight is called a **filter**(or sometimes referred to as a **neuron** or a **kernel**) and the region that it is shining over is called the **receptive field**. Now this filter is also an array of numbers (the numbers are called weights or parameters). A very important note is that the depth of this filter has to be the same as the depth of the input (this makes sure that the math works out), so the dimensions of this filter is 5 x 5 x 3. Now, let's take the first position the filter is in for example. It would be the top left corner. As the filter is sliding, or convolving**,** around the input image, it is multiplying the values in the filter with the original pixel values of the image (aka computing element wise multiplications). These multiplications are all summed up (mathematically speaking, this would be 75 multiplications in total). So now you have a single number. Remember, this number is just representative of when the filter is at the top left of the image. Now, we repeat this process for every location on the input volume. (Next step would be moving the filter to the right by 1 unit, then right again by 1, and so on). Every unique location on the input volume produces a number. After sliding the filter over all the locations, you will find out that what you're left with is a 28 x 28 x 1 array of numbers, which we call an **activation map** or **feature map**. The reason you get a 28 x 28 array is that there are 784 different locations that a 5 x 5 filter can fit on a 32 x 32 input image. These 784 numbers are mapped to a 28 x 28 array.

Activation Functions:

An activation function is a very important feature of an artificial neural network, they basically decide whether the neuron should be activated or not. They introduce non- linear properties to our Network *whose* main purpose is to convert an

input signal of a

node in an A-NN to an output signal. That output signal now can be used as an input in the next layer in the stack.

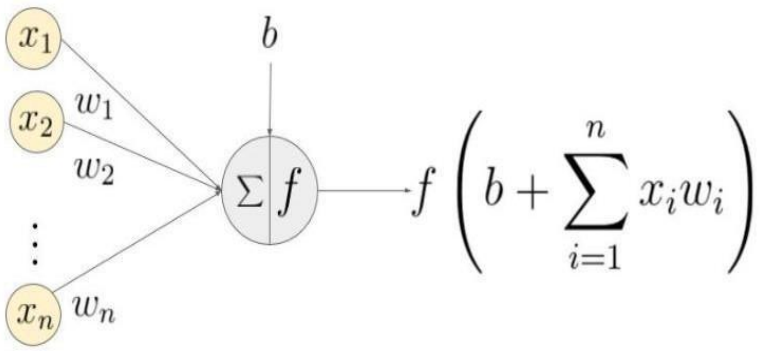


Fig 17. Working of an activation function

In the above figure, (***x1, x2 … xn***) is the input signal vector that gets multiplied with the weights (***w1, w2, …wn***). This is followed by accumulation (i.e. summation + addition of bias ***b***). Finally, an activation function ***f*** is applied to this sum.

3.4.1 ReLU (Rectified Linear Unit) Activation Function

The ReLU is the most used activation function in the world right now

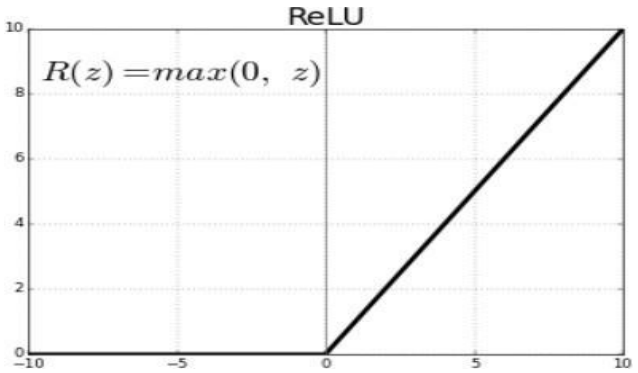


Fig 18. ReLU Activation Function

Equation: f(x) = max (0,

x) Range: (0 to infinity)

Pros:

1. The function and its **derivative** both are **monotonic**.
2. Due to its functionality it does not activate all the neuron at the same time

3. It is efficient and easy for computation.

Cons:

1. The outputs are not zero centred similar to the sigmoid activation function
2. When the gradient hits zero for the negative values, it does not converge towards the minima which will result in a dead neuron while back propagation.

3.4.2 Softmax Activation Function

The softmax function is also a type of sigmoid function but it is very useful to handle classification problems having multiple classes.

$$\sigma(z)_j = \frac{e^{z_j}}{\sum_{k=1}^{K} e^{z_k}}$$

The softmax function is shown above, where z is a vector of the inputs to the output layer (if you have 10 output units, then there are 10 elements in z). And again, j indexes the output units, so j = 1, 2, …, K.

The softmax function is ideally used in the output layer of the classifier where we are trying to attain the probabilities to define the class of each input**.**

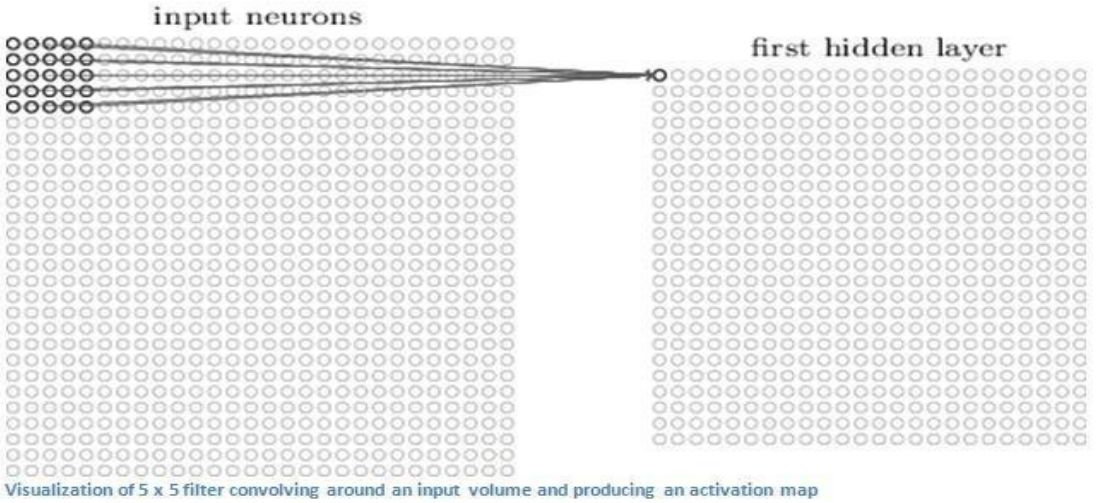


Fig 19. Visualization of 5x5 filter

3.5. Training

The way the computer is able to adjust its filter values (or weights) is through a training process called **backpropagation**.

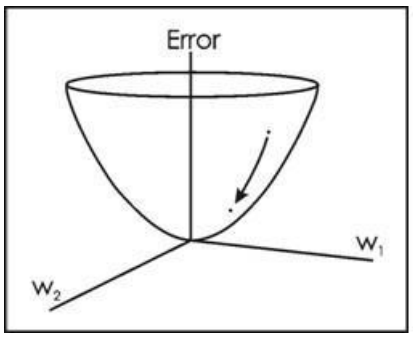
Before we get into backpropagation, we must first take a step back and talk about what a neural network needs in order to work. At the moment we all were born, our minds were fresh. We didn't know what a cat or dog or bird was. In a similar sort of way, before the CNN starts, the weights or filter values are randomized. The filters don't know to look for edges and curves. The filters in the higher layers don't know to look for paws and beaks. As we grew older however, our parents and teachers showed us different pictures and images and gave us a corresponding label. This idea of being given an image and a label is the training process that CNNs go through. Before getting too into it, let's just say that we have a training set that has thousands of images of dogs, cats, and birds and each of the images has a label of what animal that picture is. Back to backprop.

So backpropagation can be separated into 4 distinct sections, the forward pass, the loss function, the backward pass, and the weight update. During the **forward pass**, you take a training image which as we remember is a 32 x 32 x 3 array of numbers and pass it through the whole network. On our first training example, since all of the weights or filter values were randomly initialized, the output will probably be something like [.1
.1] particular. The network, with its current weights, isn't able to look for those low level features or thus isn't able to make any reasonable conclusion about what the classification might be. This goes to the **loss function** part of backpropagation. Remember that what we are using right now is training data. This data has both an image and a label. Let's say for example that the first training image inputted was a 3. The label for the image would be [0 0 0 1 0 0 0 0 0 0]. A loss function can be defined in many different ways but a common one is MSE (mean squared error), which is ½ times (actual
- predicted) squared.

$$E_{total} = \sum 1/2(target - output)^2$$

Let’s say the variable L is equal to that value. As you can imagine, the loss will be extremely high for the first couple of training images. Now, let’s just think about this

intuitively. We want to get to a point where the predicted label (output of the ConvNet) is the same as the training label (This means that our network got its prediction right).In order to get there, we want to minimize the amount of loss we have. Visualizing this as just an optimization problem in calculus, we want to find out which inputs (weights in our case) most directly contributed to the loss (or error) of the network.



One way of visualizing this idea of minimizing the loss is to consider a 3-D graph where the weights of the neural net (there are obviously more than 2 weights, but let's go for simplicity) are the independent variables and the dependent variable is the loss. The task of minimizing the loss involves trying to adjust the weights so that the loss decreases. In visual terms, we want to get to the lowest point in our bowl shaped object. To do this, we have to take a derivative of the loss (visual terms: calculate the slope in every direction) with respect to the weights.

Fig 20. 3-D Graph of Neural Net

This is the mathematical equivalent of a **dL/dW** where W are the weights at a particular layer. Now, what we want to do is perform a **backward pass** through the network, which is determining which weights contributed most to the loss and finding ways to adjust them so that the loss decreases. Once we compute this derivative, we then go to the last step which is the **weight update**. This is where we take all the weights of the filters and update them so that they change in the opposite direction of the gradient.

$$W = W_i - \eta\,(dL/dW)$$

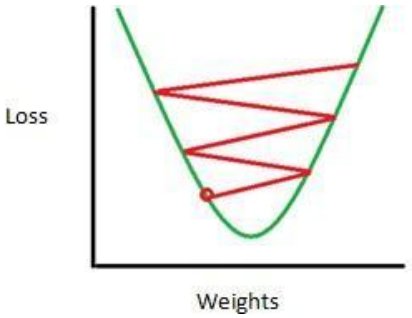
where W = Weight

W_i = Initial Weight

η = Learning Rate

The **learning rate** is a parameter that is chosen by the programmer. A high learning rate means that bigger steps are taken in the weight updates and thus, it may take less time for the model to converge on an optimal set of weights. However, a learning rate that is too high could result in jumps that are too large and not precise enough to reach

the optimal point.



Consequence of a high learning rate where the jumps are too large and we are not able to minimize the loss.

The process of forward pass, loss function, backward pass, and parameter update is one training iteration. The program will repeat this process for a fixed number of iterations for each set of training images (commonly called a batch). Once you finish the parameter update on the last training example, hopefully the network should be trained well enough so that the weights of the layers are tuned correctly.

Testing:

Finally, to see whether or not our CNN works, we have a different set of images and labels (can't double dip between training and test!) and pass the images through the CNN. We compare the outputs to the ground truth and see if our network works!

3.6. Flow of System:

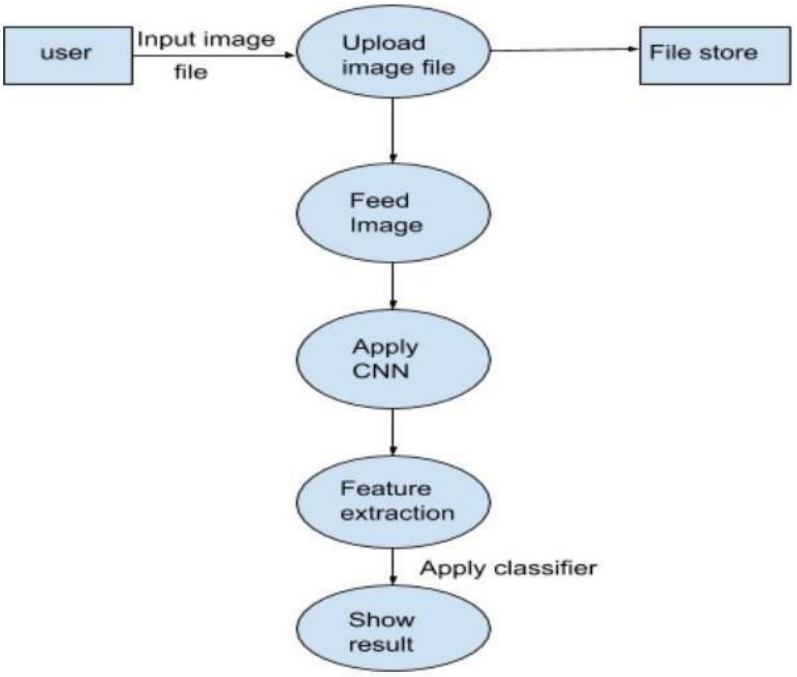


Fig 21. Flow of System

The above figure represents the actual flow of the proposed system. To develop such a system a trained dataset is required to classify an image. The trained dataset consists of train results and test results. Whenever a user will upload an input file, the image is temporarily stored in the database. This input file is then fed to the system and given to CNN where CNN is coupled with a trained dataset. CNN consists of various convolutional layers. Various features such as head, color, body, shape, and the entire image of the bird are considered for classification to yield maximum accuracy. Each alignment is given through a deep convolutional network to extract features out from multiple layers of the network. Then an unsupervised algorithm called deep learning using CNN is used to classify that image.

Further, a grey scale method is used to classify the image pixel by pixel. These features are then aggregated and forwarded to classifier. Here, the input will be compared

against

the trained dataset to generate possible results. During classification, an autograph is generated which consist of nodes that ultimately forms a network. On basis of this network, a score sheet is generated and with the help of score sheet output will be produced.

4.EXPERIMENTAL ANALYSIS AND RESULTS

4.1. System Configuration

4.1.1 Software Requirements:

Programming Language : Python

Operating System : Windows or Linux
Tools : Anaconda Navigator,Tensorflow,Keras
TENSORFLOW:

The standard name for Machine Learning in the Data Science industry is TensorFlow. It facilitates building of both statistical Machine Learning solutions as well as deep learning through its extensive interface of CUDA GPUs. The most basic data type of TensorFlow is a tensor which is a multi-dimensional array.

It is an open-source toolkit that can be used for build machine learning pipelines so that you can build scalable systems to process data. It provides support and functions for various applications of ML such as Computer Vision, NLP and Reinforcement Learning. TensorFlow is one of the must-know tools of Machine Learning for beginners.

KERAS:

Keras is an open-source neural network library that provides support for Python. It is popular for its modularity, speed, and ease of use. Therefore, it can be used for fast experimentation as well as rapid prototyping. It provides support for the implementation of convolutional neural networks**,** Recurrent Neural Networks as well as both. It is capable of running seamlessly on the CPU and GPU. Compared to more widely popular libraries like TensorFlow and Pytorch, Keras provides user-friendliness that allows the users to readily implement neural networks without dwelling over the technical jargon.

How to install Keras :

Before we go ahead with installing Keras, let us look at the installation of TensorFlow.

TensorFlow installation:

Mainly, there are two stable TensorFlow versions:

TensorFlow: for running on CPU.

TensorFlow-GPU: for running on GPU

Although with the GPU we can make very heavy computations without interrupting the work of the CPU, we are going to install the version for CPU for being simpler and is the ideal one to take the first steps with TensorFlow.
There are also the same versions with the postfix nightly, which are unstable versions in development but include the latest updates and developments.

So, to install the stable release of TensorFlow we have to write like this in our terminal:

$ pip install TensorFlow

Installing **Keras** on Python:

Installing Keras is no different from installing any other library in Python:

$ pip install keras

By default, while installing Keras, Keras will use TensorFlow as its tensor manipulation library

4.1.2 Hardware Requirements:

Processor : Intel Multicore Processor (i3 or i5 or i7)

RAM : 4GB or Above

Hard Disk : 100GB or Above

4.2. Sample Code

1.System Training :

```
import numpy as np

import pandas as pd

from keras.preprocessing.image import ImageDataGenerator, load_img

from keras.utils import to_categorical

import matplotlib.pyplot as

plt import random

FAST_RUN = False

IMAGE_WIDTH=128

IMAGE_HEIGHT=128

IMAGE_SIZE=(IMAGE_WIDTH, IMAGE_HEIGHT)

IMAGE_CHANNELS=3 # RGB color

from keras.models import Sequential

from keras.layers import Conv2D, MaxPooling2D, Dropout, Flatten,
Dense, Activation, BatchNormalization

model = Sequential()

model.add(Conv2D(32, (3, 3), activation='relu',
input_shape=(IMAGE_WIDTH, IMAGE_HEIGHT,
IMAGE_CHANNELS)))

model.add(BatchNormalization())

model.add(MaxPooling2D(pool_size=(2, 2)))
```

```
model.add(Dropout(0.25))
model.add(Conv2D(64, (3, 3), activation='relu'))
model.add(BatchNormalization())
model.add(MaxPooling2D(pool_size=(2, 2)))
model.add(Dropout(0.25))
model.add(Conv2D(128, (3, 3), activation='relu'))
model.add(BatchNormalization())
model.add(MaxPooling2D(pool_size=(2, 2)))
model.add(Dropout(0.25))
model.add(Flatten())
model.add(Dense(512, activation='relu'))
model.add(BatchNormalization())
model.add(Dropout(0.5))
model.add(Dense(200, activation='softmax'))
model.compile(loss='categorical_crossentropy', optimizer='rmsprop',
metrics=['accuracy'])
from keras.callbacks import EarlyStopping, ReduceLROnPlateau
earlystop = EarlyStopping(patience=10)
learning_rate_reduction = ReduceLROnPlateau(monitor='val_acc',
                                            patience=2,
                                            verbose=1,
```

```
                                    factor=0.5,
                                    min_lr=0.00001)
callbacks = [earlystop, learning_rate_reduction]
train_datagen = ImageDataGenerator(
    rotation_range=15,
    rescale=1./255,
    shear_range=0.1,
    zoom_range=0.2,
    horizontal_flip=True,
    width_shift_range=0.1,
    height_shift_range=0.1
)
test_datagen = ImageDataGenerator(rescale=1./255)
x_train=train_datagen.flow_from_directory(r'C:\Users\user\Desktop\datas
et\traindata',
                                          target_size = (128, 128),
                                          batch_size = 32,
                                          class_mode = 'categorical')
x_test=test_datagen.flow_from_directory(r'C:\Users\user\Desktop\dataset\
t estdata',
                                    target_size = (128, 128),
                                    batch_size = 32,
```

```
                                    class_mode = 'categorical')
print(x_train.class_indices)


model.fit_generator(x_train,
                    steps_per_epoch =130 ,
                    epochs = 15,
                    validation_data =
                    x_test,
                    validation_steps = 20,callbacks=callbacks)
model.save("mybird.h5")
from keras.models import load_model
```

Matplotlib.pyplot:

Matplotlib.pyplot is a collection of command style functions that make matplotlib work like MATLAB. Each pyplot function makes some change to a figure: e.g., creates a figure, creates a plotting area in a figure, plots some lines in a plotting area, decorates the plot with labels, etc. In matplotlib.pylot various states are preserved across function calls, so that it keeps track of things like the current figure and plotting area, and the plotting functions are directed to the current axes (please note that "axes" here and in most places in the documentation refers to the axes part of a figure and not the strict mathematical term for more than one axis).

Model = Sequential ():

The sequential model is a linear stack of layers. You can create a sequential model by passing a list of layer instances to the constructor:

```
from keras.models import Sequential
from keras.layers import Dense, Activation
```

```
model = Sequential([
```

```
    Dense(32, input_shape=(784,)),
    Activation('relu'),
    Dense(10),
    Activation('softmax'),
])
```

You can also simply add layers via the .add() method:

```
model = Sequential()
model.add(Dense(32, input_dim=784))
model.add(Activation('relu'))
```

You can save a model built with the Functional API into a single file. You can later recreate the same model from this file, even if you no longer have access to the code that created the model.

This file includes:

- The model's architecture
- The model's weight values (which were learned during training)
- The model's training config (what you passed to compile), if any
- The optimizer and its state, if any (this enables you to restart training where you left)

Batch Normalization ():

One possible reason for this difficulty is the distribution of the inputs to layers deep in the network may change after each mini-batch when the weights are updated. This can cause the learning algorithm to forever chase a moving target. This change in the distribution of inputs to layers in the network is referred to the technical name "*internal covariate shift*."

Batch normalization is a technique for training very deep neural networks that standardizes the inputs to a layer for each mini-batch. This has the effect of stabilizing the learning process and dramatically reducing the number of training epochs required to train deep networks.

Conv2D ():

This layer creates a convolution kernel that is convolved with the layer input to produce a tensor of outputs. If use_bias is True, a bias vector is created and added to the outputs. Finally, if activation is not None, it is applied to the outputs as well.

ReduceLROnPlateau ():

Reduce learning rate when a metric has stopped improving. Models often benefit from reducing the learning rate by a factor of 2-10 once learning stagnates. This call back monitors a quantity and if no improvement is seen for a 'patience' number of epochs, the learning rate is reduced.

Dropout ():

Dropout is a technique where randomly selected neurons are ignored during training. They are “dropped-out” randomly. This means that their contribution to the activation of downstream neurons is temporally removed on the forward pass and any weight updates are not applied to the neuron on the backward pass.

ImageDataGenerator ():

Data augmentation is what Kera’s’ “ImageDataGenerator” class implements. Using this type of data augmentation, we want to ensure that our network, when trained, sees new variations of our data at each and every epoch.

1. An input batch of images is presented to the ImageDataGenerator.
2. The ImageDataGenerator transforms each image in the batch by a series of random translations, rotations, etc.
3. The randomly transformed batch is then returned to the calling function.

2.Extraction of Images:

```
from keras.models import load_model
import numpy as np
import cv2
model= load_model('mybird.h5')
model.compile(optimizer='adam',loss='categorical_crossentropy',metrics=[
'accuracy'])
from skimage.transform import resize
def detect(frame):
    try:
        img = resize(frame,(128,128))
        img=np.expand_dims(img,axis=0)
        if(np.max(img)>1):
            img= img/255.0
        prediction =
        model.predict(img)
        print(prediction)
        prediction_class=
        model.predict_classes(img)
        print(prediction_class)
    except AttributeError:
```

```
        print("Shape not

        found")
from matplotlib.pyplot import *
```

```
frame=cv2.imread(r'C:\Users\user\Desktop\dataset\testdata\002.Painted_Bunting\Painted_Bunting_0091_15198.jpg')#path of image

imshow(frame)

data= detect(frame)

frame=cv2.imread(r'C:\Users\user\Desktop\dataset\testdata\009.Brewer_Blackbird\Brewer_Blackbird_0112_2340.jpg')#path of image

imshow(frame)

data= detect(frame)

frame=cv2.imread(r'C:\Users\user\Desktop\dataset\testdata\050.Eared_Grebe\Eared_Grebe_0088_34067.jpg')#path of image

imshow(frame)

data= detect(frame)
```

CV2:

OpenCV, which is an image and video processing library with bindings in C++, C, Python, and Java. OpenCV is used for all sorts of image and video analysis, like facial recognition and detection, license plate reading, photo editing, advanced robotic vision, optical character recognition, and a whole lot more.

imRead ():

cv2.imread() method loads an image from the specified file. If the image cannot be read (because of missing file, improper permissions, unsupported or invalid format) then this method returns an empty matrix.

imShow ():

cv2.imshow() method is used to display an image in a window. The window automatically fits to the image size.

3.gui.py:

```
import numpy as np
from keras.preprocessing import image
from tkinter import *
from PIL import ImageTk, Image
from tkinter import filedialog
import os
from keras.models import load_model
classifier = load_model(r'C:\Users\user\Desktop\Birds-identification-master\mybird.h5')
classifier.compile(optimizer = 'adam', loss = 'categorical_crossentropy', metrics = ['accuracy'])
root = Tk()
root.geometry("550x300+300+150")
root.resizable(width=True, height=True)
def openfn():
    filename = filedialog.askopenfilename(title='open')
    return filename
def open_img():
    x = openfn()
    test_image = image.load_img(x, target_size = (128, 128))
```

```
test_image = image.img_to_array(test_image)
```

```
    test_image = np.expand_dims(test_image, axis = 0)
    result = classifier.predict_classes(test_image)
    print(result)

index=['002.Painted_Bunting','009.Brewer_Blackbird','050.Eared_Grebe']
    label = Label( root, text="Prediction : "+index[result[0]])
    label.pack()
    img = Image.open(x)
    img = img.resize((250, 250), Image.ANTIALIAS)
    img = ImageTk.PhotoImage(img)
    panel = Label(root, image=img)
    panel.image = img
    panel.pack()
btn = Button(root, text='open image', command=open_img).pack()
root.mainloop()
```

tKinter:

The tkinter package (“Tk interface”) is the standard Python interface to the Tk GUI toolkit. Both Tk and tkinter are available on most Unix platforms, as well as on Windows systems. (Tk itself is not part of Python; it is maintained at Active State.)

Running python -m tkinter from the command line should open a window demonstrating a simple Tk interface, letting you know that tkinter is properly installed on your system, and also showing what version of Tcl/Tk is installed, so you can read the Tcl/Tk documentation specific to that version.

Label ():

The tkinter label widgets can be used to show text or an image to the screen. A label can only display text in a single font. The text can span multiple lines. We can put any text in a label and we can have multiple labels in a window (just like any widget can be placed multiple times in a window).

Button ():

The Button widget is used to add buttons in a Python application. These buttons can display text or images that convey the purpose of the buttons. You can attach a function or a method to a button which is called automatically when you click the button.

mainloop ():

There is a method known by the name mainloop () is used when your application is ready to run. mainloop () is an infinite loop used to run the application, wait for an event to occur and process the event as long as the window is not closed.

4.3. Screenshots

When we run the code, the dataset is divided into training and testing. First, it trains the system then it tests for accuracy then it asks for user input. It gives the label as a result with an input image. Following are the steps of execution:

INPUT:

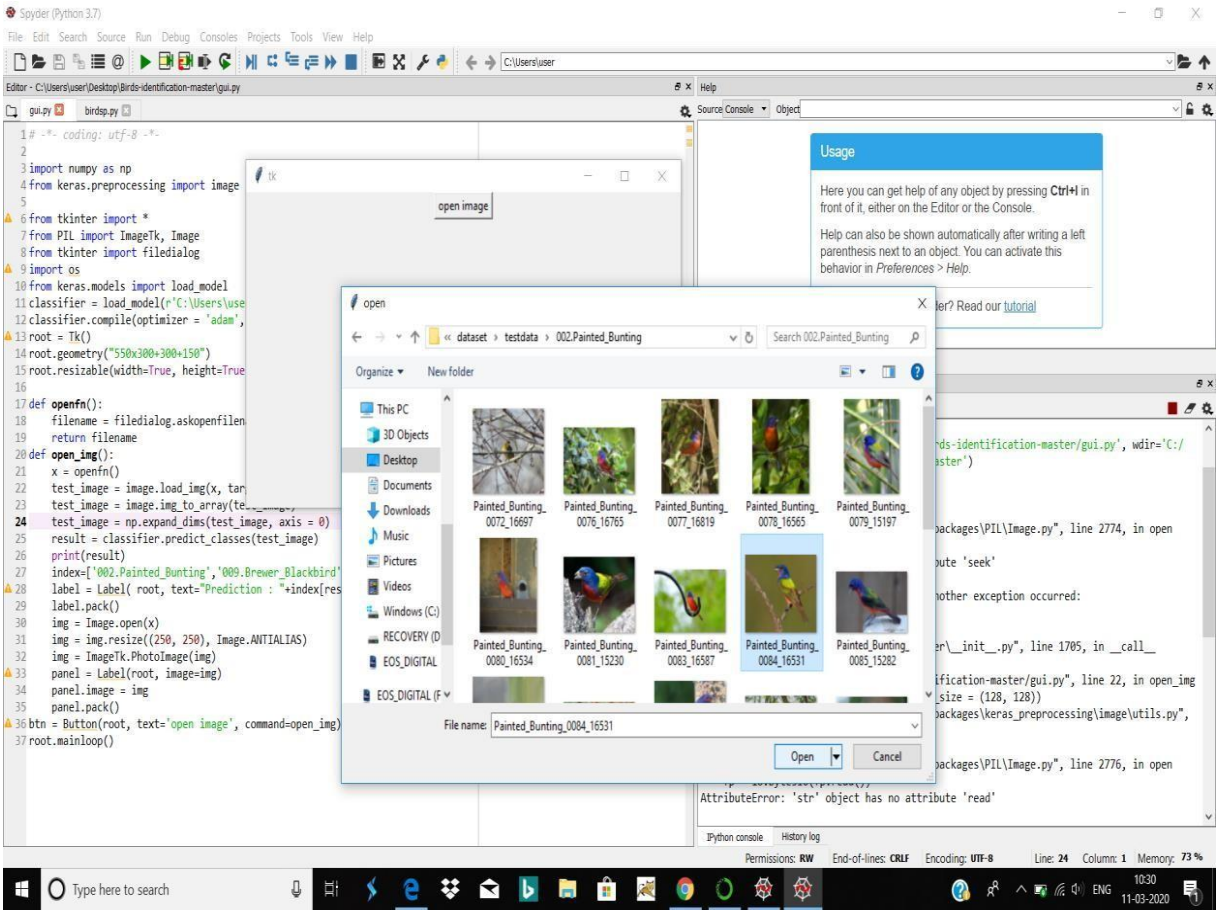
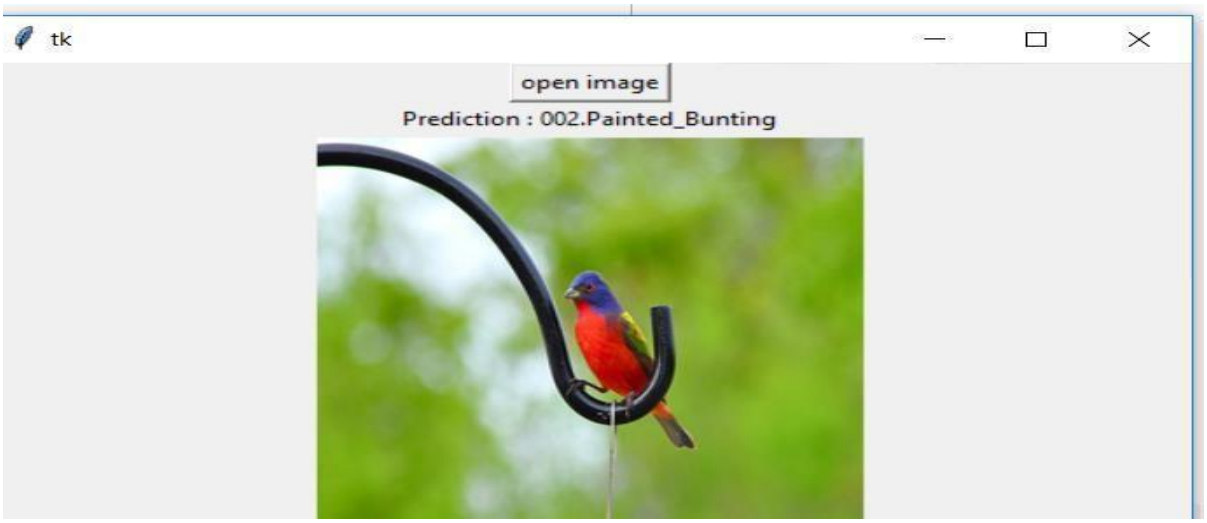


Fig 22. Inserting User Input

OUTPUT:

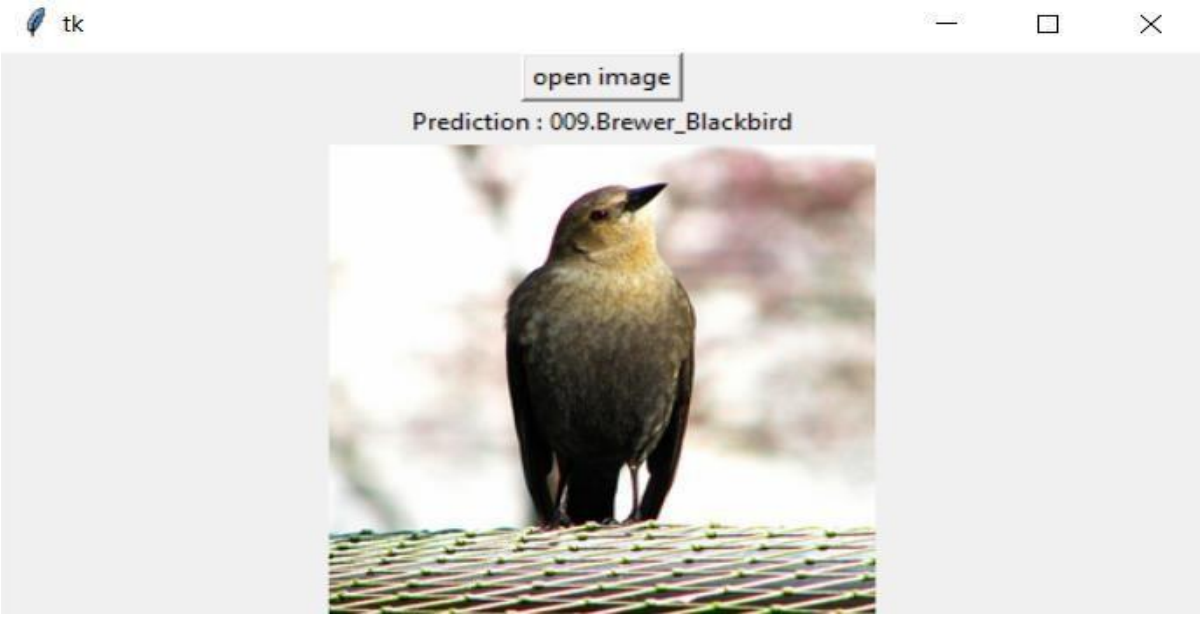
1. Output-Species Name and Input Image : Painted_Bunting

Here first we uploaded the image of Painted_Bunting of different images as an input and gone through CNN classification to obtain the name of the bird. Finally it gives us the bird name with image as an output.



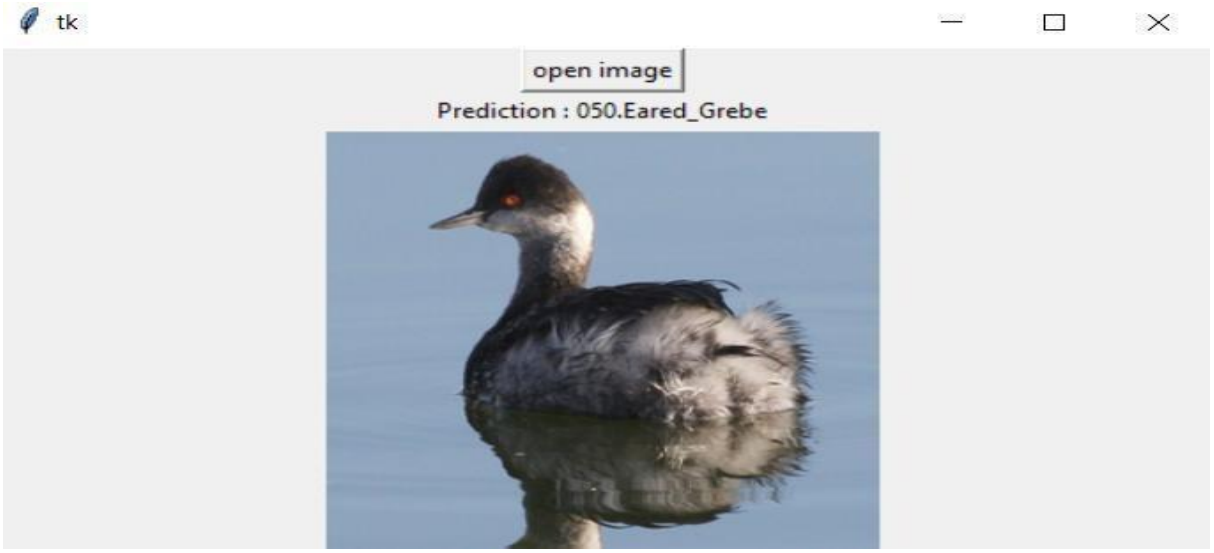
2. Output-Species Name and Input Image :Brewer_Blackbird

Here we uploaded the image of Brewer_Blackbird of different images as an input and gone through CNN classification to obtain the name of the bird. Finally it gives us the bird name with image as an output.



3. Output-Species Name and Input Image :Eared_Grebe

Here we uploaded the image of Eared_Grebe of different images as an input and gone through CNN classification to obtain the name of the bird. Finally it gives us the bird name with image as an output.



4.3. Experimental Analysis/Testing

The evaluation of the proposed approach for bird species classification by considering color features and parameters such as size, shape, etc. of the bird on the Caltech-UCSD Birds 200 (CUB-200-2011) dataset. This is an image dataset annotated with 200 bird species which includes 11,788 annotated images of birds where each image is annotated with a rough segmentation, a bounding box, and binary attribute annotations. In this the training of dataset is done by using Google-Collab, which is a platform to train dataset by uploading the images from your local machine or from the Google drive.

After training labeled dataset is ready for classifiers for image processing. There are probably average 200 sample images per species are included in dataset of 3 species which are directly captured in their natural habitat hence also include the environmental parameters in picture such as grass, trees and other factors. Here bird can identify in their any type of position as main focus is on the size, shape and color parameter. That is the image converted into number of pixels by using gray scale method, where value for each pixel is created and value-based nodes are formed which also referred as neurons. These neurons relatively defined the structure of matched pixels is simply like graph of connected nodes.

According to the nodes formed the autograph is generated which understandable by Tensorflow to classify the image. This autograph is then taken by classifiers and image is compared with the pre trained dataset images of Caltech UCSD.

Tested Results:

Sr. No	Bird name (species)	Tested Results	Accuracy (%)
1.	Painted_Bunting	10/10	95%
2.	Brewer_Blackbird	10/10	95%
3.	Eared_Grebe	7/10	70%
	Total	**27/30**	**90%**

Table 2. Efficiency Table

5.CONCLUSION AND FUTURE WORK

5.1. Conclusion:

The main idea behind developing the identification website is to build awareness regarding bird-watching, bird and their identification, especially birds found in India. It also caters to the need of simplifying the bird identification process and thus making bird-watching easier. The technology used in the experimental setup is Convolutional Neural Networks (CNN). It uses feature extraction for image recognition. The method used is good enough to extract features and classify images.

The main purpose of the project is to identify the bird species from an image given as input by the user. We used CNN because it is suitable for implementing advanced algorithms and gives good numerical precision accuracy. It is also general-purpose and scientific. We achieved an accuracy of 85%-90%. We believe this project extends a great deal of scope as the purpose meets. In wildlife research and monitoring, this concept can be implemented in-camera traps to maintain the record of wildlife movement in specific habitat and behaviour of any species.

5.2. Future Work:

1) Create an android/iOS app instead of website which will be more convenient to user.

2) System can be implemented using cloud which can store large amount of data for comparison and provide high computing power for processing (in case of Neural Networks).

REFERENCES

[1] Indian Birds [Online]https://play.google.com /store /apps /details ?id = com.kokanes.birds info & hl = en

[2] Bird Watching Apps: Five Useful Apps to Get Started With Birding [Online] https://gadgets.ndtv.com/apps/features/bird-watching-apps-five-useful-apps-to-get-started-with-birding-1640679

[3] Transfer learning using Alex Net [Online] https://in.mathworks.com /help /nnet /examples / transfer-learning-using- alexnet.html

[4] Feature extraction using AlexNet [Online] https://in.mathworks.com /help /nnet /examples / feature-extraction-using-alexnet.html # d119e4167

[5] Dipta Das, Sourya & Kumar, Akash" An Exploration of Computer Vision Techniques for Bird Species Classification", December 15, 2017.

[6] N. Dalal and B. Triggs, "Histograms of oriented gradients for human detection," IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR), San Diego, CA, 2005.

[7] C. Wah et al. The Caltech-UCSD Birds-200-2011 Dataset. Tech. rep. CNS-TR-2011-001. California Institute of Technology, 2011.

[8] Thomas Berg, Jiongxin Liu et. Al " Large-scaleFine-grainedVisualCategorizationofBirds",

[9] Yuning Chai Electrical Engineering Dept. ETH Zurich,Victor Lempitsky Dept. of Engineering Science University of Oxford, Andrew Zisserman Dept. of Engineering Science University of Oxford BiCoS:A BiSegmentation Method for Image Classification.

[10] Tóth, B.P. and Czeba, B., 2016, September. Convolutional Neural Networks for Large-Scale Bird Song Classification in Noisy Environment. In CLEF (Working Notes) (pp. 560-568).

[11] Fagerlund, S., 2007. Bird species recognition using support vector machines. EURASIP Journal on Applied Signal Processing, 2007(1), pp.64-64.

[12] U.D.Nadimpalli, R.R.Price, S.G.Hall, and P.Bomma," A Comparison of image processing techniques for bird recognition", Biotechnology Progress, Vol. 22, no. 1,pp. 9-13,2006.

[13] Toth, B.P. and Czeba, B.,2016, September. Convolutional Neural Networks for Large-Scale Bird Song Classification in Noisy Environment. In CLEF (Working Notes) (pp. 560-568).

[14] Elias Sprengel, Martin Jaggi, Yannic Kilcher, and Thomas Hofmann. Audio Based Bird Species Identification using Deep Learning Techniques. 2016.

[15] Jaderick P. Pabico, Anne Muriel V. Gonzales, Mariann Jocel S. Villanueva, and Arlene a. Mendoza. Automatic identification of animal breeds and species using bioacoustics and artificial neural networks.arXiv preprint, pages 1-17, 2015.

[10] T. S. Brandes, "Automated sound recording and analysis techniques for bird surveys and conservation," Bird Conservation International, Vol. 18, pp.v 163-173, 2008.

[11] Stefan Kahl, Thomas Wilhelm-Stein, Hussein Hussein, Holger Klinck, Danny Kowerko, Marc Fitter, and Maximilian Eibl Large-Scale Bird Sound Classification using Convolutional Neural Networks.

[12] Daniel T. Blumstein, Daniel J. Mennill, Patrick Clemins, Lewis Girod, Kung Yao, Gali Patricelli, Jill L. Deppe, Alan H. Krakauer, Christopher Clark, Kathryn A. Cortopassi, Sean F. Hanser, Brenda Mccowan, Andreas M. Ali, and ALexander N G Kirsch. Acoustic monitoring in terrestrial environments using microphone arrays: applications, technological considerations, and prospectus. Journal of Applied Ecology, 48(3):758-767, 2011.

[13] R. Bardeli, D. Wolff, F. Kurth, M. Koch, K.-H. Tauchert, and K.-H. Frommolt, "Detecting bird songs in a complex acoustic environment and application to bioacoustic monitoring," Patt Recog Letters, vol. Vol.31, pp. Pp. 1524-1534, 2010. 64.

[15] Yann LeCun, Yoshua Bengio, and Hinton Geoffrey. Deep learning. Nature Methods, 13(1):35-35, 2015.